Symbolic Modelling and Simulation of Wheeled Vehicle Systems on Three-Dimensional Roads

by

William Bombardier

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Systems Design Engineering

Waterloo, Ontario, Canada, 2009

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

In recent years, there has been a push by automotive manufacturers to improve the efficiency of the vehicle development process. This can be accomplished by creating a computationally efficient vehicle model that has the capability of predicting the vehicle behavior in many different situations at a fast pace. This thesis presents a procedure to automatically generate the simulation code of vehicle systems rolling over three-dimensional (3-D) roads given a description of the model as input.

The governing equations describing the vehicle can be formulated using either a numerical or symbolical formulation approach. A numerical approach will reconstruct numerical matrices that describe the system at each time step. Whereas a symbolic approach will generate the governing equations that describe the system for all time. The latter method offers many advantages to obtaining the equations. They only have to be formulated once and can be simplified using symbolic simplification techniques, thus making the simulations more computationally efficient.

The road model is automatically generated in the formulation stage based on the single elevation function (3-D mathematical function) that is used to represent the road. Symbolic algorithms are adopted to construct and optimize the non-linear equations that are required to determine the contact point. A Newton-Raphson iterative scheme is constructed around the optimized non-linear equations, so that they can be solved at each time step. The road is represented in tabular form when it can not be defined by a single elevation function.

A simulation code structure was developed to incorporate the tire on a 3-D road in a symbolic computer implementation of vehicle systems. It was created so that the tire forces and moments that appear in the generalized force matrix can be evaluated during simulation and not during formulation. They are evaluated systematically by performing a number of procedure calls. A road model is first used to determine the contact point between the tire and the ground. Its location is used to calculate the tire intermediate variables, such as the camber angle, that are required by a tire model to evaluate the tire forces and moments.

The structured simulation code was implemented in the DynaFlexPro software package by creating a linear graph representation of the tire and the road. DynaFlexPro was used to analyze a vehicle system on six different road profiles performing different braking and cornering maneuvers. The analyzes were repeated in MSC.ADAMS for validation purposes and good agreement was achieved between the two software packages. The results confirmed that the symbolic computing approach presented in this thesis is more computationally efficient than the purely numerical approach. Thus, the simulation code structure increases the versatility of vehicle models by permitting them to be analyzed on 3-D trajectories while remaining computationally efficient.

Acknowledgements

First I would like to thank my supervisor Dr. John McPhee for his guidance and support in my research. I would also like to thank Dr. Chad Schmitke for his advice and much help with my research. Dr. McPhee and Dr. Schmitke have done extensive investigation in this field and without their work this research would have not been possible.

I would like to also thank Dr. Bruce Minaker from the University of Windsor who encouraged me to pursue graduate studies and introduced me to the Motion Research Group at the University of Waterloo.

Furthermore, I would like to thank the Natural Sciences and Engineering Research Council of Canada (NSERC) and the University of Waterloo for the financial support on this research.

I would also to like to thank Kiumars Jalali, Tom Uchida, Willem Petersen, Matt Millard, Sukhpreet Sandhu, Chandrika Prakash, Ramin Masoudi, Akram Abdel-Rahman, Mohammadreza Saeedi, Mike Boos, Roel Vos, Étienne Genoud, Joydeep Banerjee, and Mohammad Sharif Shourijeh (members of the Motion Research Group at the University of Waterloo) who have assisted me in my research and who have made the work environment pleasant.

Moreover, I would like to thank Ahmad Fadel for taking the time to go through my thesis.

Thank you Anna-Lina for all of your support, encouragement, and patience with me.

Lastly, I would like to thank my family who has given me support and encouragement in my life. If it was not for you, I would not be who I am today nor would I have accomplished what I have so far in my life.

Dedication

This thesis is dedicated to Anna-Lina and my family who have supported and encouraged me in my studies.

Contents

List of Tables

List of Figures

Nomenclature

a	Scalar variable
$\hat{a}$	Unit vector
$\vec{a}$	Vector variable
$\dot{a}$	First derivative of a variable with respect to time
$\ddot{a}$	Second derivative of a variable with respect to time
$\mathbf{a}$	Column matrix of scalars
$\underline{\mathbf{a}}$	Column matrix of vectors (may contain scalar quantities)
$\mathbf{A}$	Matrix of scalars
$\lvert\,\rvert$	Magnitude of a variable
$\{\}$	Column matrix of variables (or ordered list of variables)
$[\,]$	Matrix of variables (or elements of a matrix)

Chapter 1

Introduction

1.1 Background

Simulation and analysis play a key role in the phases of modern product design and development, and is becoming more important since it has the capability of improving the efficiency of these phases [31]. Modifications can be done to a prototype much faster through simulation than by constructing the physical system and running tests on it. Also, the cost associated with modelling the system is much lower than performing physical tests on it.

A multibody system can be analyzed by performing three steps:

- *Model*: Describe the physical system
- *Formulate*: Generate the governing equations that describe the model
- *Simulate*: Solve the governing equations to obtain the response of the system

There are two different representations that can be used to describe the system. The first type is an acausal model, where a representation of the physical system is created using idealized components such as rigid bodies, joints (revolute, prismatic, etc.), massless springs, resistors, etc [81]. The second type is a causal model, where a set of mathematical equations is used to represent the physical system. The latter type is commonly used in control applications. In this thesis the term model is used to indicate an acausal model. The Modelling step is arguably the most important one since the other two steps are dependent on the description of the model. In creating a representation of the model, the user has to determine which characteristics are important in the model and which ones can be neglected [81].

The Formulation step involves generating the governing equations based on the description of the idealized model as input. It is during this step that the differential and algebraic equations that describe the system are constructed and prepared for simulation. The equations can be formulated using a numerical or a

symbolical approach [89]. This step is the primary interest of the work presented in this thesis.

The final step involves solving the governing equations or a portion of them based on a set of initial conditions. There are several different types of analysis that can be performed on the system based on solving the governing equations. For instance a modal analysis or an equilibrium analysis can be performed on the system equations. The two most common types of analysis in multibody dynamics are forward dynamic and inverse dynamic simulation. An inverse dynamic simulation involves solving for the forces and moments that are necessary to generate a prescribed motion, whereas a forward dynamics simulation involves numerically integrating the governing equations to obtain the motion of the system at discrete time instances. In this thesis, simulation will refer to forward dynamics simulation being performed on the governing equations.

There exists different criteria that can be used to measure the effectiveness of a model. One of the criteria is based on the capabilities of the model to determine the response of the system in different scenarios. A model is more effective if it can be used to predict the response of the system in different cases by simply modifying the set of initial conditions and parameters used to describe the model.

In vehicle dynamics for instance, it is more efficient if the model is capable of predicting the behavior of the vehicle in different situations, such as, braking, accelerating, and cornering. It is also more efficient if it can be used to analyze the vehicle in these situations as it is rolling over different road trajectories. Furthermore, the model is effective if it has the capability to analyze the ride quality of the vehicle and to determine the durability of the components of the vehicle.

Another measure of the effectiveness of a model is how quick the governing equations can be simulated to predict the behavior of the system, i.e. the simulation time. Minimizing the computational cost associated with evaluating the equations is important since it will reduce the overall time to analyze the system. It is much better to be able to analyze the system at a faster pace so that more designs can be created/modified on the system in a shorter period of time.

Simulation plays a key role in modern product design and development, especially in the automotive industry. Design changes on the vehicle can be evaluated in a fraction of the time with a computer model of the vehicle than having to run physical tests and perform data analysis on the actual vehicle. Real-time applications are becoming important since vehicles are being equipped with more safety features, such as electronic stability control. In real-time applications it is important that the model predict the behavior of the vehicle before it occurs so that it can be corrected [82]. Thus, computational cost associated with evaluating the governing equations of vehicle systems is an important measure on the vehicle model.

There are several different vehicle models that exist that can be used to analyze vehicle systems. The models are usually a compromise between the computational cost of solving the governing equations and the fidelity of the models. A high-fidelity vehicle model is one that includes all of the sub-components (the suspension

systems, the brake system, the steering system, the drivetrain, and the engine), as well as a complete finite element model of the tire and the road. However, a model with all of these components will take a long time to simulate and is far from being real-time applicable. A real-time applicable model is one that has the capabilities of obtaining the response of the system faster than real-time (actual time). Therefore, many analysts have developed simplified vehicle models that can be used to analyze the vehicle in complex situations accurately without creating physical models of all of the sub-components of the vehicle [81, 5]. One such model is the 14 DOF vehicle model proposed by Sayers and Han [81].

There are a few different commercial software packages that are specifically designed to analyze vehicle systems based on a simplified model. The governing equations are hard-coded into the program so that they do not have to be re-formulated. Therefore, the analyst only has to specify the model parameters (mass and inertia parameters, dimensions of components, center of mass locations, etc.) and the initial conditions of the system. He/she does not have the option to modify the topology in order to analyze any other vehicle system. Thus, if the analyst wishes to model a different vehicle then he/she will have to use a more generic multibody dynamics software package. The analyst will have to describe the model and formulate the governing equations. There are two approaches that can be used to formulate the governing equations, a numerical approach and a symbolical approach.

A numerical formulation approach will construct numerical matrices that describe the dynamics of the system at a specific instant in time [48]. Since the matrices are numerical in nature, they will have to be re-formulated at each time step before they are solved. There a few generic multibody software packages that are based on a numerical formulation approach, MSC.ADAMS and DADS by LMS for example. These software packages are not computationally efficient and will always generate a large set of nonlinear differential and algebraic equations (DAEs) that require iterative methods to solve them [82]. A computationally efficient model is one that simulates the system at a lower simulation time.

A symbolic formulation approach will generate symbolic mathematical expressions of the nonlinear DAEs that describe the system for all time. Therefore, the equations only need to be formulated once resulting in a reduction in the simulation time because the equations only have to be solved, not formulated, at each time step. Since symbolic mathematical expressions are obtained they can be greatly simplified using symbolic simplification techniques and optimization methods before they are solved [89]: trigonometric reductions, simplifications, and reductions can be performed on the equations; repeated terms can be determined and calculated once instead of each time they appear in the equations; and terms multiplied by 0 can be removed [102]. Furthermore, the equations can be viewed so that the analyst can get some physical insight from them. They are also portable since a mathematical expression of the governing equations is obtained so that they can be formulated using one software package and solved using another. Thus, symbolic formulations offer many advantages over numerical ones.

There exists a major disadvantage to a symbolic formulation approach and this is the tendency of the equations to grow large as the complexity of the model increases [39]. The computer memory required to manage the symbolic equations can become so large that the computer can not handle it anymore and comes to a halt. However, intelligent formulation methods have been created that can resolve this problem [39, 89]. An intelligent formulation method is an approach that allows the governing equations to be formulated in a manner that permits them to be solved more effectively by a numerical integrator [39].

One of the challenges to creating a multibody dynamics model of the vehicle is the incorporation of the highly non-linear tire since modern tire models have properties that require specific treatment during all three of the analysis stages [81, 70]. This challenge has been overcome by Sayers and Han [81] and by Morency [70] by defining specific steps and tools to follow when incorporating the tire in a multibody dynamics model of the vehicle. Morency [70] has created an approach to analyze the vehicle using a symbolic formulation that can be incorporated in a generic multibody dynamics software package that uses a symbolic formulation method. However the approach defined by Morency [70] is limited to vehicles rolling on a flat road. Therefore, there is a need to develop a method to analyze vehicle systems on three-dimensional (3-D) roads using a symbolic formulation approach to improve the capabilities of vehicle models and to benefit from the many advantages of a symbolic formulation.

1.2 Research Contributions

The major research contribution of this work is the extension of the steps outlined by Sayers and Han [81] to include a tire on a 3-D road in multibody systems model of the vehicle. This includes the development of the steps and equations necessary to determine the contact point between the tire and the ground. Also with the development of the road model that is based on the thin disk tire model with variable radius to locate the contact point which was developed by Postiau et al. [76].

Another research contribution of this work is the creation of a simulation code structure that permits the pneumatic tire and the road to be incorporated within a symbolic formulation procedure of vehicle systems. The simulation code structure consists of a number of procedure calls that are used to determine the magnitude and the direction of the tire forces and moments that act on the vehicle model. The inclusion of a tire and a road component with an approach that formulates the governing equations symbolically permits the automatic generation of simulation code that can be used to analyze vehicle systems on 3-D trajectories.

The first part of the simulation code structure involves an external function call to the road model to determine the contact point between the tire and the ground. The determination of this point is dependent on the road profile, and

thus the external function is also dependent on the road profile. The second major contribution of this work involves the automatic creation of the road model based on the mathematical function used to define the 3-D road. The external function is automatically created in the formulation stage as part of the simulation code structure using symbolic tools from the general purpose computer algebra software package. This permits the analyst to only have to define the road profile as an elevation (3-D mathematical) function when the model is created.

The principle advantage of the equations that are used to determine the contact point between the tire and the ground and the simulation code structure is that they can be implemented in any method that is used to generate the governing equations using a symbolic formulation approach. Thus, the many advantages of symbolic formulation can be utilized to analyze vehicle systems more efficiently. The ultimate goal of this research is to extend the capabilities of vehicle models while benefiting from the advantages of symbolic formulation.

1.3 Structure of Thesis

This thesis consists of 6 chapters. The first chapter introduces the research topic and describes the motivation and goals behind it. The second chapter is a literature review on the necessary literature required to achieve the goals of the research. The chapter discusses the essentials to modelling multibody dynamic systems and in particular vehicle systems. It also goes through the different software packages that are available to analyze general multibody dynamic systems as well as vehicle systems. Since a major part of the research involves tire and road modelling, a literature review is also given on these topics.

Chapter 3 describes how the tire on a 3-D road can be implemented in a multibody dynamic model of the vehicle. The theory and steps to accomplish this are described in detail followed by a discussion of how the contact point between the tire and the ground is determined. The thin disk tire model with variable radius is introduced along with the methods that were taken to improve the model. The Newton-Rapshon iterative scheme is also introduced as the method to solve the two equations necessary to determine the point of contact. The chapter ends with describing the different ways of defining the road.

Chapter 4 discusses how to implement the approach in a computer environment. The custom Newton-Raphson iteration method that was created to locate the contact point is presented along with improvements that were made to the custom approach. The hardware float environment is discussed followed by the presentation of the way to automatically generate the road model based on the custom Newton-Raphson method with the road profile as input. Simulation code optimization is discussed along with how optimization methods were used to optimize the road model. The simulation code structure is also presented in conjunction with a complete discussion of each of the sections of the structure. The chapter ends with

a discussion on how the structure is implemented in the DynaFlexPro commercial software package as a tire/road component.

The tire/road component is justified in chapter 5 by comparing simulation results obtained from DynaFlexPro and from MSC.ADAMS for different vehicle maneuvers on different road profiles. The vehicle model is presented along with modelling aspects used in both MSC.ADAMS and DynaFlexPro. Three examples are presented for the Chevrolet Equinox model performing braking and steering maneuvers on different road profiles that can be defined by a single elevation function. Three more examples are presented with the Equinox performing braking and steering maneuvers on roads that can not be defined by a single elevation function, and thus have to be defined in tabular form. The examples show that there is excellent agreement between DynaFlexPro and MSC.ADAMS, thus justifying the approach. They also confirm that significant gains in the simulation time are achieved with a symbolic formulation approach as compared to the purely numerical one. Furthermore, the results confirm the capabilities of the model to analyze the vehicle on different road profiles since 2 out of the 6 examples could not be simulated in MSC.ADAMS.

Chapter 6 concludes the thesis by giving a summary of the accomplishments of the research, and ends with a discussion of the areas for future research.

Chapter 2

Literature Review

2.1 Multibody Dynamics

2.1.1 Modelling Equations

The single pendulum will be used throughout this chapter for example purposes, and is shown in Figure (2.1). The forces in the free body diagram of the pendulum

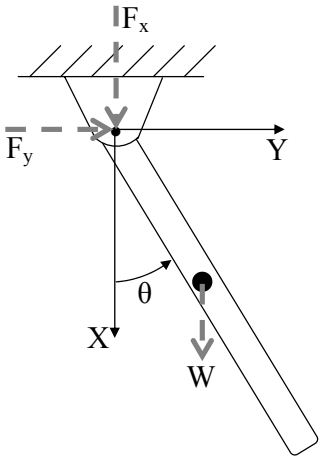


Figure 2.1: Single Pendulum

are also shown in the picture as the dashed arrows. A free body diagram is a diagram of the system that shows all forces and moments acting upon it [51]. The pendulum is two-dimensional (2-D), but the general three-dimensional (3-D) approach will be used to analyze the pendulum since this general method is applicable to 2-D problems.

The first step to obtain the modelling equations of multibody systems is to determine the number of degrees of freedom (DOF) of the system, which is the

minimum number of coordinates needed to completely describe the configuration of the system [19]. The number of DOF, f, of a system can be determined by subtracting the number of constraints, m, in the system from the number of coordinates, n, of the system as is shown in Equation (2.1) [19].

$$f = n - m \tag{2.1}$$

The configuration of each body in the system is determined by a set of three position coordinates and three orientation coordinates. Thus, 6 DOF are required to determine the configuration of a body floating freely in cartesian space. The constraints associated with the system are what restricts the movement of the bodies in space. Each constraint will decrease the number of DOF by one. The pendulum in Figure (2.1) has only one degree of freedom (f = 6-5 = 1). The revolute joint at the top of the pendulum introduces 5 constraints on its movement. Thus, if the angle θ is known for all time, then the position and orientation of the pendulum would be completely specified for all time.

The analyst has the option to choose the variables he/she would prefer to completely describe the system. The variables associated with the position and orientation of the bodies of the system are referred to as generalized coordinates and the variables chosen to represent the velocity and angular velocity are referred to as generalized speeds. This is the notation used by Ginsberg and Baruh and it will be adopted in this thesis [44, 19].

There are a number of methods that can be used to obtain the equations of motion of multibody systems in terms of the generalized coordinates and speeds and some of the most popular methods are as follows: Newton-Euler equations, Lagrange's equations, principle of virtual work, Hamilton's principle, D'Alembert's principle, and linear graph theory [19, 44, 65]. The Newton-Euler method is the oldest and is still commonly used to obtain the equations of motion. The main disadvantage with the approach is that a free body diagram (FBD) is needed in order to obtain the equations of motion. All of the other methods are analytical methods and do not require an FBD to obtain the equations.

The Newton-Euler equations are based on Newton's second law of motion in both the translational and rotational domains [44], and are shown in Equations (2.2) and (2.3).

$$\sum \vec{F} = m\dot{\vec{v}} \tag{2.2}$$

$$\sum \vec{M} = \mathbf{I_G}\dot{\vec{\omega}} + \vec{\omega} \times \mathbf{I_G}\vec{\omega} \tag{2.3}$$

where m is the mass of the body, $\mathbf{I_G}$ is the inertia dyadic, $\vec{v}$ is the velocity of the body, and $\vec{\omega}$ is the angular velocity. The equations of motion are determined for a multibody system by applying the Newton-Euler equations to each body in the system. A detailed example of how the Newton-Euler equations are used to obtain the dynamic equations of a simplified example is shown in Appendix (A) to

assist the reader in understanding how this approach can be utilized to obtain the equations.

The orientation of bodies in cartesian space is determined based on a rotation matrix that relates the orientation of the body frame to another (usually taken as the global reference frame) [44]. The rotation matrix can not be represented by a vector because rotations between two frames are not commutative. Therefore, another set of rotational coordinates is needed to track the orientation of bodies in cartesian space. A set of three rotations (either body fixed or space fixed rotations) can be used to accomplish this [44]. Each rotation is by a specified angle about an axis and these angles are commonly known as Euler angles. The 3-2-1 (Z-Y-X) body fixed Euler angle sequence will be used in this thesis to locate the orientation of bodies in space.

A state space representation of the governing equations introduces additional equations that express the derivatives of the generalized coordinates in terms of the generalized speeds in order to reduce the order of the dynamic equations [19]. The Newton-Euler equations are written in terms of accelerations and angular accelerations; thus the derivative of the generalized speeds will appear in the dynamic equations as is shown in Equations (2.2) and (2.3) [83, 88]. Let the additional equations be denoted as kinematic differential equations. In this thesis, the kinematic differential equations are represented by the generalized speeds as direct derivatives of the generalized coordinates. Thus, the state space representation will be a combination of the dynamic equations and the kinematic differential equations to give a set of first order ordinary differential equations (ODEs). Also, let the generalized coordinates and generalized speeds that appear in the state space representation be denoted state variables.

Consider the single pendulum and suppose the analyst chooses to model the pendulum using three coordinates: the x position of the center of mass, the y position of the center of mass, and the angle θ. Thus, the generalized coordinates are x, y, and θ, and let the generalized speeds be v_x, v_y, and ω_z, where v_x is the the x component of the velocity of the center of mass, v_y is the y component of the velocity of the center of mass, and ω_z is the angular velocity of the pendulum about the z axis. The state space representation of the dynamic equations is shown in Equations (2.4)-(2.9). The dynamic equations are obtained using the Newton-Euler equations and are shown in Equations (2.4)-(2.6), and the kinematic differential equations are shown in Equations (2.7)-(2.9).

If the number of generalized coordinates chosen to model the system is greater than the number of degrees of freedom of the system, then the coordinates are not independent. If this is the case, then a set of constraint equations (algebraic equations) is needed to express the dependencies of the chosen generalized coordinates. There are two types of constraint equations: holonomic and non-holonomic. Holonomic constraint equations only involve position level variables. On the other hand, non-holonomic constraint equations involve non-integrable relationships between the generalized speeds [47]. The constraint equations in this thesis will be

restricted to that of holonomic type.

$$\dot{v_x} = \frac{W + F_x}{m} \tag{2.4}$$

$$\dot{v_y} = \frac{F_y}{m} \tag{2.5}$$

$$\dot{\omega}_z = F_x \frac{6}{ml} \sin(\theta) - F_y \frac{6}{ml} \cos(\theta) \tag{2.6}$$

$$\dot{x} = v_x \tag{2.7}$$

$$\dot{y} = v_y \tag{2.8}$$

$$\dot{\theta} = \omega_z \tag{2.9}$$

where m is the mass of the pendulum, l is the length of the pendulum that is assumed to be a slender rod, W is its weight in the gravitational field, and F_x and F_y are the reaction forces at the revolute joint.

For the single pendulum, two independent constraint equations are required that relate the three chosen coordinates because there is only 1 DOF. The two constraint equations are shown in Equations (2.10) and (2.11).

$$x = \frac{l}{2} \cos(\theta) \tag{2.10}$$

$$y = \frac{l}{2} \sin(\theta) \tag{2.11}$$

Therefore, there is a set of 8 equations (Equations (2.4)-(2.11)) with 8 unknowns $(v_x, v_y, \omega_z, x, y, \theta, F_x, F_y)$ that represent the system. This set of equations can be solved to obtain the response of the system.

The dynamic equations and the constraint equations need to be modified into a form that is more suitable for simulation. From this point forward it will be always assumed that a state space representation will be used to represent the dynamic equations and they will be specified in terms of the accelerations for simplification purposes. Also, let the constraint equations along with the dynamic equations be denoted the governing equations. Haug [47] proposed a matrix form that represents the governing equations and is shown in Equations (2.12) and (2.13).

$$\mathbf{M}(\mathbf{q}, t)\, \ddot{\mathbf{q}} + \mathbf{\Phi}_{\mathbf{q}}^T \boldsymbol{\lambda} = \mathbf{F}(\mathbf{q}, \dot{\mathbf{q}}, t) \tag{2.12}$$

$$\mathbf{\Phi}(\mathbf{q}, t) = 0 \tag{2.13}$$

where $\mathbf{M}$ is the generalized mass matrix, $\mathbf{q}$ is a column vector of the generalized coordinates, $\mathbf{\Phi}_{\mathbf{q}}^T \lambda$ is a column vector representing the effect of the reaction forces on the system, and $\mathbf{F}$ is the generalized force matrix which contains the effects of the external forces and quadratic speed terms (centripetal and Coriolis). The general form of the dynamic equations is shown in Equation (2.12) and the general form of the algebraic constraint equations is shown in Equation (2.13), where one side of the equations is set to zero.

The governing equations representing the pendulum can be rearranged into Haug's matrix form as shown in Equations (2.14) and (2.15).

$$\begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & \frac{m}{12}l^2 \end{bmatrix} \begin{Bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{\theta} \end{Bmatrix} + \begin{Bmatrix} -F_x \\ -F_y \\ -F_x \frac{l}{2}\sin(\theta) + F_y \frac{l}{2}\cos(\theta) \end{Bmatrix} = \begin{Bmatrix} W \\ 0 \\ 0 \end{Bmatrix} \quad (2.14)$$

$$\begin{Bmatrix} x - \frac{l}{2}\cos(\theta) \\ y - \frac{l}{2}\sin(\theta) \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix} \quad (2.15)$$

The constraint equations are normally non-linear algebraic equations in terms of the generalized coordinates [47]. However, the velocities and accelerations can be obtained by solving sets of linear equations that are in terms of the known position variables [47]. Taking the derivative of Equation (2.13) with respect to time results in Equation (2.16) that is linear in terms of $\dot{\mathbf{q}}$.

$$\mathbf{\Phi_q}\dot{\mathbf{q}} = -\mathbf{\Phi}_t \quad (2.16)$$

Differentiating the equation again with respect to time results in Equation (2.17) which is linear in terms of $\ddot{\mathbf{q}}$.

$$\mathbf{\Phi_q}\ddot{\mathbf{q}} = -(\mathbf{\Phi_q}\dot{\mathbf{q}})_\mathbf{q}\dot{\mathbf{q}} - 2\mathbf{\Phi}_{\mathbf{q}t}\dot{\mathbf{q}} - \mathbf{\Phi}_{tt} = \boldsymbol{\gamma} \quad (2.17)$$

Equations (2.16) and (2.17) are the velocity-level and acceleration-level constraint equations. The matrix $\mathbf{\Phi_q}$ that appears in both equations as well as in the dynamic equations is the jacobian matrix. It is a matrix of partial derivatives that is related to $\mathbf{\Phi}$ and the generalized coordinates, and is determined using Equation (2.18).

$$\mathbf{\Phi}_{\mathbf{q}[i,j]} = \frac{\partial \mathbf{\Phi}_{[i]}}{\partial \mathbf{q}_{[j]}} \quad (2.18)$$

Note, the other terms in the two equations are also partial derivative terms that are calculated in a similar fashion as the jacobian matrix.

The jacobian matrix for the single pendulum is shown in Equation (2.19).

$$\mathbf{\Phi_q} = \begin{bmatrix} 1 & 0 & \frac{l}{2}\sin(\theta) \\ 0 & 1 & -\frac{l}{2}\cos(\theta) \end{bmatrix} \quad (2.19)$$

The transpose of the jacobian matrix is part of the dynamic equations as can be seen in Equation (2.12) and is zero when the number of generalized coordinates is equal to the number of degrees of freedom, i.e. $\mathbf{\Phi}$ is null. Multiplying this term by the column vector of Lagrange multipliers ($\boldsymbol{\lambda}$) gives the term in the dynamic equations that represents the effect the reaction forces have on the system equations as was pointed out previously. The Lagrange multipliers represent the reaction forces that are necessary to force the system constraints and in the pendulum example are F_x and F_y.

The most general form of the governing equations is a set of differential and algebraic equations (DAEs) that represent the system. The set of equations is a set that contains the $n + m$ dynamic and constraint equations. The algebra constraint equations ($\mathbf{\Phi} = 0$) are usually highly non-linear and thus require an iteration solution to solve them and this tends to be computationally expensive. Methods for solving sets of ODEs are more efficient than methods for solving sets of DAEs. This is the reason why most dynamic analysts try to eliminate the non-linear algebraic constraint equations from the set of equations that represent the system [47, 19, 44]. This can be accomplished using the acceleration-level of the constraint equations (Equation (2.17)) instead of the position-level, so that a set of ODEs is obtained. The updated governing equations are shown in Equation (2.20).

$$\begin{bmatrix} \mathbf{M} & \mathbf{\Phi}_{\mathbf{q}}^T \\ \mathbf{\Phi}_{\mathbf{q}} & 0 \end{bmatrix} \begin{Bmatrix} \ddot{\mathbf{q}} \\ \boldsymbol{\lambda} \end{Bmatrix} = \begin{Bmatrix} \mathbf{F} \\ \boldsymbol{\gamma} \end{Bmatrix} \tag{2.20}$$

These equations can be specified in state space form and solved numerically using a solver to obtain the generalized coordinates and speeds ($\mathbf{q}$ and $\dot{\mathbf{q}}$), and the Lagrange multipliers ($\boldsymbol{\lambda}$).

The main problem with using the acceleration-level of the constraint equations is that they are only satisfied at this level. A build up of numerical integration error may cause a significant violation in the position and velocity-level constraints [47]. If the position-level constraints are not satisfied then the solution will wrongfully indicate that joints in the system are floating apart.

There are different methods that can be employed to try to enforce the position-level constraints. Baumgarte constraint stabilization may be used to minimize the error in the position level constraints [20]. Baumgarte [20] added a few extra terms to Equation (2.17) so that the equation can be inherently stable (spring and damper system), which implies that $\dot{\mathbf{\Phi}} \approx \mathbf{\Phi} \approx 0$. The updated form of the acceleration-level equation is shown in Equation (2.21).

$$\mathbf{\Phi}_{\mathbf{q}}\ddot{\mathbf{q}} = \boldsymbol{\gamma} - 2\alpha(\mathbf{\Phi}_{\mathbf{q}}\dot{\mathbf{q}} + \mathbf{\Phi}_t) - \beta^2\mathbf{\Phi} = \boldsymbol{\gamma}' \tag{2.21}$$

Baumgarte constraint stabilization is implemented by replacing $\boldsymbol{\gamma}$ in Equation (2.20) with $\boldsymbol{\gamma}'$. It is difficult to choose the Baumgarte stabilization parameters (α and β) that result in an accurate solution since there is no general method for selecting the parameters [47].

Partitioning methods can also be used to decrease the error of the constraint equations. With this method the generalized coordinates are partitioned into a column vector of $\mathbf{u}$ independent coordinates and $\mathbf{v}$ dependent coordinates. It is possible to express the dynamic equations as a set of ODEs that is equal to the number of degrees of freedom (f) for any system [39, 47]. Thus, the number of independent coordinates should be equal to f, and the number of dependent coordinates should be equal to m. Fisette et al. [39] showed that the dependent velocities and accelerations ($\dot{\mathbf{v}}, \ddot{\mathbf{v}}$) can be explicitly solved and eliminated from the dynamic equations. However, since the constraint equations are non-linear at the

position-level they cannot in general be eliminated from the dynamic equations. Thus, the dynamic equations can be expressed in terms of $\mathbf{u}$, $\dot{\mathbf{u}}$, $\ddot{\mathbf{u}}$, and $\mathbf{v}$ as is shown in Equation (2.22).

$$\mathcal{M}(\mathbf{u}, \mathbf{v}, t)\,\ddot{\mathbf{u}} + \mathcal{C}(\dot{\mathbf{u}}, \mathbf{u}, \mathbf{v}, t) \;=\; \mathcal{F}(\dot{\mathbf{u}}, \mathbf{u}, \mathbf{v}, t) \tag{2.22}$$

Note, the $\mathcal{M}$ and $\mathcal{F}$ matrix are different than the ones presented in Equation (2.12). Thus, before the equations are solved an iterative technique is required to solve the constraint equations for the dependent variables based on the current values of the independent variables. From the point of view of the numerical integrator, the simulation code appears to describe a set of ODEs that are in terms of the independent variables even though the constraint equations still exist [39].

If an explicit solution exists to the constraint equations, then the solution can be used to replace the numerical iteration at each time step to solve for the dependent variables. This will result in a simulation that is more computationally efficient. Kecskemethy et al. [56] have created a method to determine whether or not the constraint equations can be solved analytically and to include the solution into the dynamic equations when it exists.

In general, there are three options that can be employed to eliminate the numerical error associated with using the acceleration-level of the algebraic constraint equations and these are as follows.

- Use an ODE solver to solve for the response of the governing equations using Equation (2.12) with Baumgarte stabilization
- Use a solver that is capable to solve DAEs directly [30, 6]
- Use an ODE solver to integrate the reduced dynamic equations (Equation (2.22)) and use a numerical iteration routine at each time step to solve the position-level constraint equations

2.1.2 Equation Complexity and Modelling Variables

The user has the possibility of choosing the modelling variables (generalized coordinates) that he/she wishes to represent the system. Each multibody system contains a number of sets of possible modelling variables that can be used to represent the system. The complexity of the system equations usually decreases as the number of modelling variables increase [19]. It was seen in Section (2.1.1) that choosing x, y, and θ as the generalized coordinates resulted in a set of 5 equations that would need to be solved to obtain the response of the pendulum. Only one generalized coordinate is required to analyze the system since there is only a single degree of freedom. Thus, the analyst could choose only θ as the generalized coordinate, which results in the single equation shown in Equation (2.23).

$$\frac{m}{3}l^2\ddot{\theta} = -mg\frac{l}{2}\sin(\theta) \tag{2.23}$$

This single equation is more complex than any of the previous 5 equations. However, the response can be obtained by solving this single equation. If the analyst still wishes to know the x and y positions of the center of mass, then the constraint equations can be used to determine their values once the solution for θ is known. This very simple example has shown that the choice of the modelling variables can have an effect on the number of equations used to represent the system and the complexity of those equations.

Since the selection of modelling variables is very important in modelling of multibody dynamic systems there is additional literature on the topic [66, 69].

There are many different types of modelling coordinates that are commonly used to model multibody dynamic systems, including absolute coordinates, joint coordinates and indirect coordinates.

Absolute coordinates are very common in numerical formulations of the governing equations [47]. In an absolute coordinate formulation, 6 generalized coordinates (3 rotations and 3 translations) are used to describe the motion of each body in the system. Thus, 6 dynamic equations would be considered for each body and a number of constraint equations would be needed to enforce the dependencies between the generalized coordinates. This results in a large number of governing equations and solving them would require numerical integrators that are capable of dealing with DAEs. Solving these equations involves obtaining the motion of the bodies and the constraint forces [91].

Joint coordinates are very common in robotic applications since the joint angles are required for most analyzes [19]. Joint coordinates describe the relative motion between two adjacent bodies; between two adjacent arms on a robot for example. There are two types of topologies, open loop and closed loop. An open-loop topology implies that there are no closed kinematic chains within the model [65]. For open loop topologies, there will be one joint coordinate for every degree of freedom in the system. Thus, a minimal number of equations is obtained with no constraint equations. On the other hand, for closed loop topologies, constraint equations will be necessary and a set of DAEs is obtained to represent the system. The equations expressed in joint coordinates are usually more complex than the equations expressed in absolute coordinates, but are fewer in number [54].

Indirect coordinates are similar to joint coordinates with the difference that they measure the relative motion between any two bodies in the system. McPhee et al. [66] have shown that for certain problems, a set of indirect coordinates can lead to less complex equations than a set of joint coordinates.

2.2 Commercial Software for Multibody Dynamics

The governing equations need to be formulated and solved to determine the response of the system. It has been one of the main objectives in multibody dynamics to

automatically formulate the governing equations given a description of the system as input [24]. There are two approaches that can be used to achieve this: a numerical approach or a symbolic approach. A numerical formulation will re-construct the governing equations at each time step, whereas a symbolic approach will formulate the governing equations ahead of time and use the result at each time step to solve the system of equations using a numerical solver.

There exists several different software packages that are designed to specifically solve multibody systems. Each package uses a different method to formulate the governing equations based on a certain set of modelling variables. Some of the most popular ones are as follows.

- MSC.ADAMS
- DADS
- ROBOTRAN
- AutoSim
- Modelica / Dymola
- DynaFlexPro

The two first software packages use a numerical formulation approach and the latter ones use a symbolic formulation approach.

2.2.1 MSC.ADAMS

Mechanical Dynamics Inc. (MDI) developed Automatic Dynamic Analysis of Mechanical Systems (ADAMS) in the late 1970s to analyze general mutlibody dynamic systems [84]. In 2002, MSC Software Corporation acquired Mechanical Dynamics Inc., and thus ADAMS is now part of MSC Software Corporation.

MSC.ADAMS is recognized as the most popular software for analyzing multibody systems. It will automatically convert a topological model to a set of dynamic equations at each time step and solve them using a numerical solver [60]. The topological model is defined graphically within the MSC.ADAMS environment and the topology is based on the components used to define the model. MSC.ADAMS can perform both kinematic and dynamic analysis as well as static equilibrium analysis. There is also no restriction in the topological interconnection between bodies. Thus open-loop, closed-loop, and multiple loop topologies can be analyzed and are all treated the same.

MSC.ADAMS is a numerical formulation approach that uses a set of absolute coordinates as modelling variables. MSC.ADAMS has a built-in library of the equations that represent each of the components in the system. The library is

based on the equations developed by Haug [47]. The equations for a rigid body will appear in the dynamic equations and the equations for all other components will appear in the constraint equations. Thus, at each time step the governing equations are re-formulated based on the topology of the system using the pre-defined equations in the library. MSC.ADAMS uses the simplified form of the governing equations (Equation (2.22)) with the Newton-Raphson iteration scheme to solve the constraint equations at each time step [57].

The major disadvantage to MSC.ADAMS is the high computational cost to simulating systems. A large set of DAEs is always obtained to represent the system and this leads to high computational demands to solve the system of equations. It is also based on a numerical formulation approach that requires the equations to be re-formulated at each time step and this also leads to more computational demands to solve the system.

2.2.2 DADS

Dynamic Analysis and Design System (DADS) is a general purpose multibody dynamics software package developed and commercialized by Haug [47]. Currently, DADS is commercialized as one of the lines of product for LMS International and is the second most widely used software package to solve multibody dynamic systems based on a numerical approach.

DADS uses a very similar approach to model dynamic systems as MSC.ADAMS. The main difference between DADS and MSC.ADAMS is the approach to which the equations are solved. MSC.ADAMS uses a numerical solver to solve a set of ODEs with a Newton-Raphson iteration scheme embedded into the solver to solve the constraint equations. On the other hand, DADS will solve a set of algebraic equations using a specialized DAE solver. Note, MSC.ADAMS does offer a variety of solvers, but the default solver is the one specified above.

The DAE solver used by DADS will first reduce the dynamic equations to state space form. Euler's numerical integration method is then used to convert the set of first order ODEs to a set of algebraic equations. The equations are then combined with the algebraic constraint equations to create a system of equations. A numerical solver can then be used to solve the system of algebraic equations. This specialized solver permits the constraint equations to be satisfied automatically, but has the disadvantage that it relies on the numerical error of Euler's method [48].

2.2.3 ROBOTRAN

ROBOTRAN was developed by Fisette et al. [39] to automatically generate the governing equations of mechanical systems given a description of the system as input. It is written in C with all of the symbolic operators being hard-coded by the developers. Thus, it is a standalone program. The mathematical models obtained

in ROBOTRAN can be exported into a different form (Fortran, C, or Matlab routines) allowing them to be simulated within another environment [39]. It is currently implemented in the Matlab/Simulink environment to analyze multibody systems to permit the ease of analyzing the model [13].

ROBOTRAN uses a symbolic formulation approach that is based on joint coordinates as the set of modelling variables to model the system. The coordinate partitioning method is used to minimize the numerical error in the constraint equations. Thus the reduced set of governing equations are used to to represent the system (Equation (2.22)). The governing equations are generated based on d'Alembert's potential power principle [39] based on the library of symbolic operations that is already hard-coded. The algebraic constraint equations are solved at each time step using an iterative scheme and an ODE solver is used to solve the dynamic equations.

Laurent Sass [80] developed an algorithm to analyze electromechanical systems that is based on the symbolic approach that is utilized in ROBOTRAN. Symbolic models are generated separately for both the electrical and mechanical domains and then combined to formulate the global symbolic model [80]. The global model is then used by a numerical integrator to determine the response of the system using the same method that is used by ROBOTRAN [80]. The symbolic equations representing the mechanical domain are obtained using ROBOTRAN. The equations representing the electrical domain are obtain using ROBOTRAN's symbolic engine to develop Kirchoff's equations using graph theoretic algorithms [80]. The approach to obtain the electrical equations is implemented in ELECTRAN along with the method to obtain the global symbolic equations. ELECTRAN is not currently commercially available.

2.2.4 AutoSim

AutoSim was developed in the early 1990s at the University of Michigan to automatically formulate the governing equations of mechanical systems using Kane's method. Kane's method is based on D'Alembert's principle and uses the kinematic differential equations to obtain the generalized active forces and the generalized inertia forces [25]. It is a standalone program written in the Lisp language.

Heuristics are used in AutoSim to determine the set of modelling variables to represent the system. Thus, the user does not have the control over which variables he/she wishes to use to model the system. The heuristics used by AutoSim to choose the modelling variables is based on an approach that attempts to minimize the number of equations and the complexity of the equations. However, in most cases AutoSim uses a set of joint coordinates.

AutoSim also gives the user the option to identify the state variables that have a negligible effect on the dynamics of the system. Basically, trigonometric functions that are in terms of the negligible variables are replaced with a truncated Taylor series expansion and products of the negligible variables are dropped when they are

of order 2 or higher [83]. Essentially, the equations are simplified using techniques that mimic a dynamic analyst cutting out terms that he/she feels are not important in the equations.

AutoSim will also perform small modifications to the equations before they are solved. Any constant terms are computed before simulation and their values replaced inside of the equations. This eliminates parameters from having to be re-calculated at every time step resulting in a reduction in the computational time required to evaluate the expressions [83]. AutoSim also has the capability to export the equations in the form of C or Fortran code.

An ODE solver is used to solve the dynamic equations when joint coordinates are applied to represent the system. In AutoSim, joint coordinates are adopted when the system is an open-loop system, and a coordinate partitioning method is used when the system contains closed kinematic chains. Note, in this case, the modelling variables will consist of a set that usually contains a mixture of absolute and joint coordinates. At every time step, the dependent speeds are integrated to estimate the dependent coordinates for the next time step. In most cases these estimates will be inaccurate; thus a Newton-Raphson iteration scheme is used to correct them by performing a single iteration [83]. This is done to decrease the computational time associated with solving the constraint equations while still ensuring that they are satisfied.

2.2.5 Modelica — Dymola / MapleSim

The Modelica language is an object-oriented language that allows multi-domain systems to be analyzed. A multi-domain system is a system that contains combinations of mechanical, electrical, thermal, and hydraulic components. Modelica allows the interactions between different energy domains to be modelled, and is especially popular in the design and analysis of mechatronic systems.

The purpose of the Modelica language is to have a standard format that permits models in different domains to be exchanged between different tools and users [35]. Each component of the multi-domain system may contain differential, algebraic, and discrete equations, as well as internal functions, and is represented as an object. A Modelica translator is required to formulate the equations into a meaningful form and then to simulate them. The most popular translator is Dymola which is created by DynaSim and is now part of Dessault Systèmes.

Dynamic Modelling Laboratory (Dymola) uses graph-theoretic approaches to choose the modelling variables to model the system. Thus, the user does not have the option to choose them. The equations are stacked together to form the governing equations that represent the system and reduction methods are used, such as Tarjan's algorithm, to simplify the equations [37, 36]. Such a method is described by Nordström et al. [72] as an in-line integration approach. Specialized techniques are applied in Dymola to manipulate the DAEs into a form that permits them to be solved efficiently [37, 36, 26].

MapleSim has been developed by MapleSoft Inc. to analyze multi-domain systems. The majority of the components in MapleSim are based on the Modelica standard libraries [15]. The governing equations are solved in MapleSim using a specialized DAE solver.

2.2.6 DynaFlexPro

DynaFlexPro (DFP) is a commercial software package that is used to analyze multi-domain systems. It is implemented in Maple, which is a general purpose computer algebra software package. DFP was developed by McPhee and Schmitke [8] and commercialized by MotionPro Inc. before being acquired by MapleSoft Inc. DynaFlexPro uses linear graph theory to automatically formulate the governing equations symbolically based on the topological description of the system as input [8]. The governing equations are formulated using the general form of the governing equations with Baumgarte stabilization. Thus, an ODE solver is used to solve the governing equations at each time step to determine the response of the system [8].

DynaFlexPro allows the user to choose the modelling variables he/she wishes to represent the system. McPhee and Redmond [66] have shown that an analyst can choose to model the system with joint, absolute, or indirect coordinates by selecting an appropriate tree for the linear graph representation. McPhee [65] has also shown that it is possible to model the system using a hybrid form of the coordinates. DynaFlexPro also has a heuristics approach embedded into it that can be used to choose the modelling variables such that a minimal set of the governing equations is obtained or a set that is the least complex. The heuristics are based on approaches outlined by Léger [59].

Many of Maple's symbolic math capabilities are used within DynaFlexPro to improve the computational power associated with solving the governing equations. DFP uses the *simplify()* and *combine()* Maple commands to simplify the governing equations primarily using trigonometric reductions. The CodeGeneration package is applied within DynaFlexPro to optimize procedures by eliminating repetitive function calls and unnecessary calculations. It is also used to export the simulation code to a different language.

The analyst also has the option to view the governing equations within the Maple environment in order to make further simplifications to them or to perform further operations on them. Also, the user can gain some physical insight of the physics of the system by looking at the equations. Furthermore, design sensitivities can be obtained by taking the symbolic derivatives of the governing equations inside the Maple environment.

Linear Graph Theory

Linear graph theory (LGT) was invented by Leonhard Euler in 1736 and has been combined with the principles of physics to analyze multibody systems [64]. With

this approach, the topology of the physical system is represented as a collection of nodes and edges. LGT permits the freedom to choose the modelling variables to model the system through the choice of the spanning tree.

The spanning tree is a connected subgraph that includes all nodes, but does not form a closed loop. The edges in the linear graph are the components themselves, whereas the nodes are points on the graph where a connection between two components exists. The edges selected into the spanning tree are denoted as branches, and the edges not selected into the tree are denoted as chords. The chords form what is known as the cotree.

Each edge has an associated set of through and across variables associated with it. The through variables of an edge are the variables that can be measured by a device that is in series with the component (forces and moments), whereas the across variables of an edge are the variables that can be measured by a device in parallel with the component (positions, velocities, and accelerations).

An across space of an edge is an ordered set of unit vectors that span the space in which the component allows its across variables to vary. Likewise, the through space is an ordered set of unit vectors that span the space in which the edge permits its through variables to vary. The through and across spaces of an edge will be orthogonal to one another and it is important that the spanned spaces reflect the physical nature of the component [88].

Each edge in the linear graph has an associated set of terminal equations that are used to represent the physical behavior of the component. For example, the Newton-Euler equations in d'Alembert form represent the physical behavior of a rigid body and thus are considered as terminal equations to an edge representing a rigid body.

A fundamental cutset (f-cutset) is a set of edges that when removed, divides the graph into two non-connected parts and consists of a single branch and a unique set of chords. There is one f-cutset for every branch in the spanning tree. A fundamental circuit (f-circuit) equation is a set of edges that form a closed loop and contains one chord and a unique set of branches. There is one f-circuit equation for every chord in the cotree.

The principle of orthogonality is one of the approaches that can be used to formulate the governing equations using linear graph theory. With this method, the dynamic equations are formulated by projecting the fundamental cutset equations onto the across space for the edges selected into the tree [65]. Thus, the motions consistent with the across space of edges selected into the tree will become the generalized coordinates. The constraint equations are formulated by projecting the fundamental circuit equations onto the through space of the edges selected into the cotree [65]. The reaction forces and moments that are used to enforce the constraints will be consistent with the through space of the edges selected into the cotree.

The fundamental circuit equations are also used to express the chord across variables in terms of the branch across variables. Similarly, the f-cutset equations

are used to express the branch through variables in terms of the chord through variables. This is done so that the dynamic and algebraic equations are only expressed in terms of the primary variables (modelling variables).

There is plenty of literature available that explain in detail how linear graph theory is used to obtain the governing equations of multi-domain systems. Thus, for further information the reader may refer to [65, 66, 88, 89]. However, a detailed example of how LGT is used to obtain the dynamic equations of a simplified example is shown in Appendix (A) to assist the reader in understanding the essentials of the approach.

2.3 Modelling of Vehicle Systems

There are several different vehicle models that can be used to analyze vehicle systems. The models are a compromise between the fidelity one wishes to have versus the computational cost associated with formulating the governing equations of the model and solving them. High-fidelity vehicle models will include all of the components of the model in complete detail (the front and rear suspension systems, the steering system, the braking system, the drivetrain system, and the engine). These models have the capability of determining the response of the vehicle by modelling all of the components, but have a huge overhead in the time required to completely specify the model and with the computational cost in simulating the model [23].

In recent years, there have been a few different vehicle models developed to simplify the model while still maintaining the capability of predicting the behavior of the vehicle accurately [81]. These models usually consist of a torque applied directly to the tires to eliminate the motor and drivetrain and the braking system from having to be modeled. They also eliminate the complete steering system from being modeled by applying the steering motion directly to the tires. The models also use a simplified suspension system to model the front and rear suspensions. Let a model of this form be denoted as a simplified model.

Michael Sayers [81] used this simplified model to define a generic four-wheeled vehicle model with independent suspension for the specific analysis of the passenger car. Bombardier also used this simplified model to analyze the cornering performance of a multiple truck trailer system for some confidential contract work he performed while working on his MASc degree at the University of Waterloo. He et al. [49] also used this topology to model an articulated steer vehicle to determine its stability characteristics.

The four-wheeled vehicle model with independent suspension developed by Sayers is the most widely used simplified model because of its specific application to passenger cars and because of its computational performance that is feasible for real-time simulation [82, 5, 9]. The generic model is shown in Figure (2.2) [70]. The suspension system at every corner of the model is modeled as a prismatic joint with a spring and damper along the z axis of the vehicle chassis. The suspension

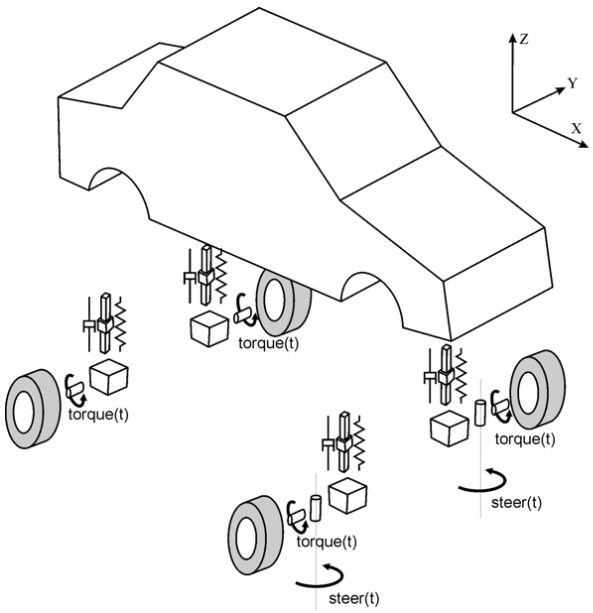


Figure 2.2: Generic 4 Wheeled Vehicle Model with Independent Suspension [70]

systems do not include anti-roll bars and the spring and damping rates are linear. Also, the wheel orientations are not dependent on the kinematics of the actual suspension system. Revolute joints (shown as cylinders in Figure (2.2)) are used to permit the wheel to rotate about its spin axis and a motion driver is used to model the steering, where the latter only applies to the front wheels. However, the steering angles will not add a degree of freedom to the model because they are prescribed motions of time. The torque from the engine (the torque require to propel the vehicle forward) and the braking torque (the torque required to stop the vehicle) are applied to the revolute joint representing the spin axis of the tire. Both the torque and the steering are applied to the model as functions of time and the same amount is used on all joints respectively. The unsprung mass of the suspension system is modeled as a rigid body between the prismatic and revolute joints (or steering motion drivers for the front corners of the vehicle).

The generic vehicle model is a 14 degree of freedom (DOF) model: 3 associated with the translation of the vehicle body and 3 associated with its orientation, 4 associated with the deflection of the suspension systems, and 4 affiliated with the wheel spins. Thus, 14 generalized coordinates can be used to model the vehicle so that no constraint equations are required. The suggested set of modelling variables is the 6 coordinates representing the vehicle body, the joint angles of the revolute joints, and the joint coordinates of the prismatic joints [81].

The fidelity of the simplified vehicle model can be increased by using lookup tables to modify the position and orientation of the wheels as a function of suspension deflection [82]. Lookup tables can also be used to provide spring and damping forces. Also, the axis of the suspension system (the axis of the prismatic joints) can

be tilted to account for the fact that the suspension deflection may not be entirely in the vertical direction [82]. Sayers [82] has also proposed a method to include the extra roll stiffness (could be from an anti-roll bar for example) in the model.

Sayers [81] used AutoSim to formulate the governing equations of the generic four wheeled vehicle model. The governing equations went through a series of modifications and some fine-tuning before they were incorporated into the commercial software package CarSim. CarSim is used world wide to design and test passenger vehicles. Sayers [81] used CarSim for hardware in the loop testing for an electronic control unit of a baking system.

ADAMS/Car RealTime from MSC.Software and ve-DYNA from Tesis DYNAware are also software packages that are used to analyze passenger cars based on the simplified model [5, 9]. These packages specifically target real-time applications of vehicle systems.

The specialized programs, such as CarSim and ve-DYNA, offer many advantages to analyzing vehicle systems, even though the topology of the vehicle is fixed. Most of them offer a graphical user interface that allows the user to enter the parameters specifically for the model and to choose certain options he/she wishes to use in the model, such as the tire model. They also offer a very good post-processing tool that has the capabilities of plotting all of the results and performing animations on them.

There are a number of other software packages that have been developed to analyze vehicle systems that are not dependent on a fixed vehicle topology. Thus, the user has the option to model any vehicle system, such as a truck trailer system or an articulated steered vehicle for example.

ADAMS/Car is a product developed by MSC.Software, and uses MSC.ADAMS to obtained the governing equations of vehicle models. In order to analyze vehicle systems in ADAMS/Car all components of the vehicle must be specified and thus the vehicle model is a high-fidelity vehicle model [4]. ADAMS/Car has been extensively used in industry to design and analyze vehicle systems. However, it is based on a complete numerical approach and is thus not feasible for real-time applications. ADAMS/Car also has the capabilities to analyze specific vehicle components, such as the suspension, by performing extensive kinematic and dynamic analysis on the system.

Postiau et al. [76] used ROBOTRAN to formulate the governing equations of a 3-D vehicle model with multi-link suspension at all wheels. They exported the simulation code into the C language and imported it into the Matlab/Simulink environment. The vehicle system was simulated in Matlab/Simulink, and a 10 second double lane change maneuver took about 11 seconds to simulate [76].

Elmqvist et al. [36] used Dymola to create vehicle models with various different topologies. One of the topologies used was a similar topology to the simplified model described by Sayers [36]. The simulation code obtained from Dymola was exported in the form of C code and compiled on a real-time target machine using software

from Matlab and Opal-RT [81]. They performed a double lane change maneuver in real-time. They also incorporated a MacPherson strut front suspension and a trailing arm rear suspension into their model. An analytical solution exists to the constraint equations associated with the MacPherson strut and a trailing arm suspension; thus they used the analytical solution to the constraint equations along with an ODE solver to solve the dynamic equations. The simulation performed in real time for a double lane change maneuver [36].

Morency [70] developed an approach to analyze wheeled vehicle systems on a flat road when a symbolic formulation approach is used to obtained the governing equations of vehicle systems. Morency [70] developed a tire component that permits the tire forces and moments to be calculated at each time step through a number of procedure calls. His approach was implemented in the DynaFlexPro (DFP) software package and is now an add-on component to DFP called DynaFlexPro/Tire.

DFP offers many advantages to analyzing vehicle systems. The topology of the system is not fixed in DFP; thus the user has the option to analyze any vehicle system. The modelling variables are not fixed in DFP allowing the user to choose the set of generalized coordinates he/she wishes to represent the system. DFP can analyze multi-domain engineering systems, and it is based in the Maple symbolic language allowing symbolic simplification and optimization techniques to be performed on the equations once they are formulated. However, it is restricted to vehicle systems rolling on a flat road.

2.4 Tire Modelling

The response of vehicle systems is highly determined from the development of the forces and moments that result from the interaction between the tire and the road [23, 81]. The only other major external forces that act on vehicle systems are the aerodynamic forces and the gravitational forces, with the aerodynamic forces having a secondary influence [81].

The interaction between the tire and the ground is dominated by the deformation of both the tire and the ground in each of the contact patches developed between the tires and the ground. There are four different possibilities to model the deformation and contact between the two bodies and they are: use a rigid tire and a rigid ground; use a deformable tire and a rigid ground; use a deformable tire and a deformable ground; and use a rigid tire and a deformable ground [17]. The latter approach is commonly used in off-road vehicle modelling since the ground will deform much more than the tire [21]. The first type is commonly used to model rail vehicles (e.g. trains) and the second type is mostly used to model passenger vehicles where the tire is a pneumatic tire and the road ia a paved road surface or a solid dirt surface [23, 103]. Since the second type is applicable to passenger vehicles, it is the type that is most widely used.

The carcass of the pneumatic tire is constructed from a metal belt with polymer

fibers woven at precise angles to the belt. Rubber compounds are coated around the carcass to increase its strength and improve its wear characteristics. The tread pattern of the tire is carved into the rubber compounds. The tire is fabricated out of highly non-linear anisotropic materials. Thus, it is difficult to predict the exact behavior of the tire using an analytical model [23]. There are also many uncontrollable factors that are difficult to predict that affect the tire forces and moments (inflation pressure and tread wear for example). Finite element methods can be used to predict the response of the tire, but there is an overhead with the computational cost associated with using these methods.

There are two different ways to model the tire and the method chosen is dependent on the type of analysis that is going to be performed on the vehicle. A ride analysis is typically done on a vehicle to perform a durability analysis on a suspension unit or to determine the ride comfort of the passengers [74]. A handling analysis is performed on the vehicle to determine its performance in several different situations such as acceleration, cornering, braking, or combinations of the three [74]. A ride analysis is usually a frequency analysis performed on the vehicle, whereas a handling analysis is an analysis where the dynamic response of the vehicle is determined.

In a ride analysis, the tire is represented as a combination of springs and dampers in the normal direction (direction normal to the road) in order to perform a vibration analysis on the tire and the vehicle to determine the natural frequencies and mode shapes of the system. A typical ride analysis performed on a passenger car is a four post test, where each of the tires is displaced vertically to determine the displacement of the chassis as well as its settling time [43]. Tire models used in this type of analysis are denoted as physical tire models.

In a handling analysis of the vehicle, the pneumatic tire can be represented as a single spring and damper in the normal direction since the vibrations in the tire are accurately predicted by this configuration [74]. However, it is difficult to determine an analytical model that can predict the lateral and longitudinal force, as well as the moments that are also generated because of the interaction of the tire with the road. Therefore, analysts rely on physical tests to predict how the tire is going to behave in certain situations. The tire is tested on a machine that measures the forces and moments at the interaction between the tire and the ground for various wheel orientations and deflections, forward velocities, and spin rates [23]. The results from the tests are then summarized in tabular form and can be used during simulation to predict the tire forces and moments. However, a large amount of interpolation will be needed to determine the proper force (or moment) to use because there is likely a large number of data points to interpolate through. This leads to very slow simulations.

Mathematical functions can be used to fit curves to the measured data. The models try to extract important characteristics from the physical tests and present them as a set of equations that can be used to predict the tire forces and moments based on a set of tire intermediate parameters as input. The set of parameters

describe the current state of the tire. These mathematical expressions are easier to evaluate as compared to interpolating through a whole table. These types of tire models are known as mathematical tire models and will be referred to as tire models in this thesis. Tire models are considered as being semi-empirical because they are based on results from test data with the structure of the equations that properly describe the behavior of the tire [75].

Tire models are most widely used in vehicle dynamic studies since they predict the lateral and longitudinal forces as well as the moments that arise in the contact patches between the tire and the road. There are several different proposed types of tire models that can be used to predict the tire forces and moments. The most accurate tire models are very complex and are based on a large set of data, such as the 2002 Pacejka tire model [74]. Simplified tire models exist that can predict the tire forces and moments based on a small set of parameters that describe the tire, such as the Fiala tire model [23]. The tire model is a compromise between the desired accuracy and computational cost of evaluating the expressions. The 2002 Pacejka tire model [74] and the Fiala tire model [38] are used in this thesis to model the tire.

2.5 Road Modelling

The road model plays an equally important role as the tire model in the calculation of the tire forces and moments. There are two generic ways to model the road: as a solid surface or as a deformed surface. The latter is commonly used in off-road vehicle modelling where the road is normally defined as a complete finite element mesh. This method has been used by Schmid [86, 85] to define the terrain in order to analyze a Military vehicle in off-road applications. Schmid [86] also showed that when the vehicle is equipped with pneumatic tires it is important to also define the tire as a finite element mesh to properly determine the contact and deformation between the tire and the road. Proper determination of the contact between the tire and the road and of the deformation of the tire leads to proper calculation of the tire forces and moments.

The most common types of road models assume that the deflection in the road is negligible. This is the most common type since the passenger vehicle is the most widely analyzed vehicle and in most circumstances is rolling on a solid surface (pavement/ashfalt, cement, solid dirt). There are multiple ways of defining the road as a solid surface, and some of the most popular ones are as follows: as a triangular mesh, as a complete finite element mesh, as a mathematical function, and as a single surface normal vector [23, 11].

The road model is defined so that the contact between the tire and the ground can be effectively determined [76]. For the longest time the road was defined as a single plane at zero elevation with the surface normal being equivalent to the global Z axis (if the global Z axis corresponds to the vertical axis of the vehicle body) [23].

This permitted the contact between the tire and the ground to be easily calculated as a single point contact. Note, it is at the center of pressure in the contact patch where the tire forces and moments act [23]. Thus a key part in determining the tire forces and moments is the determination of the contact point between the tire and the ground.

It became noticed that a flat road is not sufficient enough to completely predict the behavior of the vehicle. Thus, mathematical functions were used to model road obstacles that were superimposed on the flat road to determine the response of the vehicle in situations where it encountered a hill or a pothole [23, 11]. As finite element methods became more popular, the road was defined as a triangular surface mesh or as a complete finite element mesh [23]. The road is defined as a set of triangular patches when the triangular surface mesh is used to model the road, where each patch is modeled as a flat surface with the element surface normal being defined [11]. This permitted elevation changes in the road surface to be modeled [23]. The road is modeled as a 3-D (complete) finite element mesh in situations where both handling and ride analysis are important [74]. In these situations the tire is also modeled using finite element methods and the contact is determined based on the interaction of the two meshes [74].

The road has also been defined as a 3-D elevation function ($z = f(x, y)$) which defines the surface of the road everywhere in the cartesian space. Postiau et al. [76] used this definition to analyze a vehicle with multi-link suspension on different 3-D road trajectories. They showed that this method works best in a symbolic approach to analyzing the vehicle [76].

In the past decade, it has become more and more popular to define the road in tabular form by specifying the centerline of the road and using a piecewise cubic spline to define the road between different data points [23, 11]. This permits obstacles to be analyzed as well as a complete track to be defined allowing the response of the vehicle to be determined. The road bank angle is also defined in the table as well as the friction coefficients of the road surface. Also, with this approach a driver model is required to force the vehicle to stay on the path defined by the centerline of the road [23]. This method is used by most software packages that are specific to vehicle analysis, such as CarSim and MSC.ADAMS for example [11, 10]. The methods to define the road will be further explored in this thesis.

Chapter 3

Pneumatic Tires on a 3-D Road in a Multibody Systems Context

3.1 Overview

The response of an automobile in all situations is highly dependent on the forces developed in the contact patch between each of the tires and the ground. The tire forces developed at the tire/road interaction are what governs how the vehicle responds to external forces, such as the aerodynamic force, and to changes in the vehicle state [81].

3.2 Incorporating a Tire on a 3-D Road in a Multibody Systems Model of the Vehicle

The determination of the tire forces is important when modelling vehicle systems. Sayers and Han [81] outlined a number of steps necessary to incorporate the highly non-linear tire in a multibody dynamics model of the vehicle. However, in the development of their approach, they assumed that the road surface is flat and can not take on any three-dimensional (3-D) shape. The steps outlined by Sayers and Han can be extended to include the effects of the road by adding an intermediate step to determine the contact point between the tire and the ground [24]. The necessary steps to include a tire rolling on a 3-D road in a multibody dynamics model of the vehicle are as follows.

1. Define a point where the tire forces will act on the multibody model of the vehicle
2. Determine an expression to locate the contact point between the tire and the ground

3. Select an expression for the normal force

4. Establish directions for the ISO unit vectors

5. Determine expressions for the most common kinematic inputs required by tire models

6. Use a tire model to determine expressions for the magnitudes of the remaining tire forces and moments

The second step (2) is the additional step that is required to analyze vehicle systems on 3-D roads since the location of the contact point is not known a priori.

The six steps will be discussed in detail, followed by an explanation of the thin disk tire model with variable radius — the chosen method to calculate the contact point between the tire and the ground. Details of the approach are given, followed by a discussion of how the contact point is determined. The chapter ends with a breakdown of how to solve the two equations that determine the contact point at each time step.

3.2.1 Define a Point Where the Tire Forces Will Act on the Multibody Model of the Vehicle

The forces that result from the interaction of the tire and the road act at the contact point between the tire and the ground (point P in Figure (3.1)). The contact point is the center of pressure of the contact patch (contact area), which is necessary to transmit the forces and moments between the tire and the ground. It is not a fixed point on the tire nor on the road, thus it is inconvenient to apply the forces and moments at this point. Therefore, it is common practice to apply them at the center of the tire (point C in Figure (3.1)) [23, 76, 81, 96].

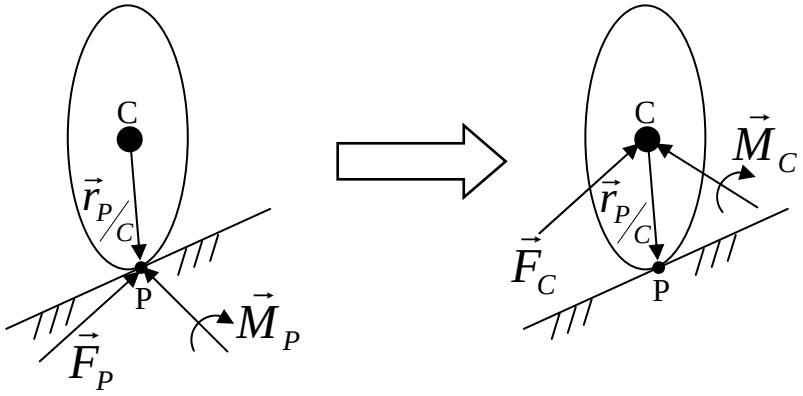


Figure 3.1: Point of Force Application of the Tire Forces and Moments

An additional moment needs to be considered when moving the point of force application to the tire center. This additional moment is dependent on the vector from the tire center to the contact patch ($\vec{r}_{P/C}$).

$$\vec{F_C} = \vec{F_P} \tag{3.1}$$

$$\vec{M_C} = \vec{M_P} + \vec{r}_{P/C} \times \vec{F_P} \tag{3.2}$$

3.2.2 Determine an Expression to Locate the Contact Point Between the Tire and the Ground

The contact point between the tire and the ground is easily determined when the road is assumed to be a flat road. In this particular case, the point of contact is located at the geometric center of the contact patch [23] (point P in Figure (3.3)). There exists a few methods that can be used to calculate the vector $\vec{r}_{P/C}$ when the road is assumed to be flat. Two of the most popular methods assume a shape of the tire to determine this vector. The first method assumes the tire can be approximated as a thin disk and was proposed by Postiau and Sayers [76, 81]. The second method was recommended by Blundell and Harty and approximates the tire as a toroid [23]. In practice, both methods give very similar results for small camber angles and either one can be used [74]. However, for larger camber angles, as can be seen in motorcycles for example, the second method is more accurate [74].

Either of these methods can be used to determine the vector $\vec{r}_{P/C}$ when the road is assumed to take on any 3-D form for small camber angles. The first approach will be used in the present work because of its ease of implementation. However, when the road is assumed to vary, the contact point can not be assumed to be at the geometric center of the contact patch and its location has to be determined.

In the present work, the thin disk tire model with variable radius will be used to determine the point of contact, and is shown in Figure (3.2). With this approach the tire is assumed to be a planar disk with variable radius. Essentially, the radius of the disk is allowed to vary along with its location in the wheel plane until the tire comes into contact with the ground. The radius of this disk is the loaded radius (R_l) and the vector $\vec{r}_{P/C}$ is the vector going from point C to point P in the wheel plane of the thin disk. Refer to section (3.3) for more details on this approach.

3.2.3 Select an Expression for the Normal Force

A tire model is used to calculate the forces and moments in the contact patch and one of the most common inputs to tire models is the normal force [23]. The normal force will act in the same direction as the road surface normal at the point of contact. The vehicle model is developed to analyze vehicle systems in acceleration, braking and handling situations. Therefore, the tire can be modeled as a single spring and damper in the normal direction, as shown in Figure (3.3), since the frequency of oscillation of the tire is captured accurately using this model [95].

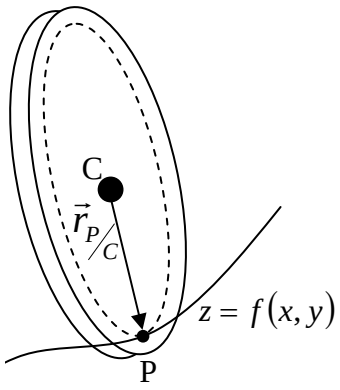


Figure 3.2: Contact Point Determination

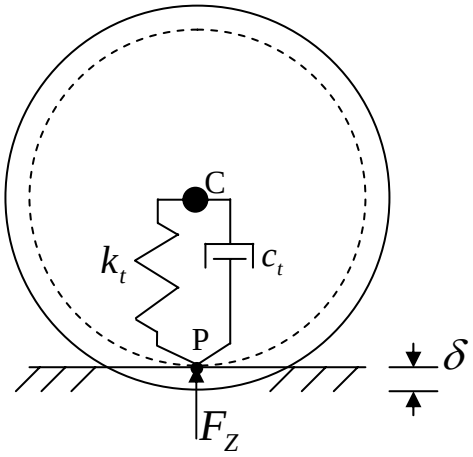


Figure 3.3: Calculation of the Tire Normal Force

With this approach, the normal force (F_Z) is given by Equation (3.3).

$$F_Z = (max)((k_t\delta + c_t\dot{\delta}), 0) \tag{3.3}$$

where, δ represents the tire penetration into the ground and the maximum function (max) is used to ensure that the normal force is zero when the tire is no longer in contact with the ground, k_t is the spring constant and c_t is the damping coefficient. If the road contains small obstacles that are close to the size of the contact patch, then a more advanced method is required to calculate the normal force [23, 74].

3.2.4 Establish Directions for the ISO Unit Vectors

A tire model will give the magnitudes of the tire forces and moments and a tire axis system will specify in which direction they act. Both are needed since a force

is a vector and needs a magnitude and a direction. There are two popular tire axis systems — the Society of Automotive Engineers (SAE) tire axis system, and the International Organization for Standardization (ISO) tire axis system [43]. The latter system is shown in Figure (3.4), and will be adopted in the present work since it is a more common system, and since SAE is considering adopting the system.

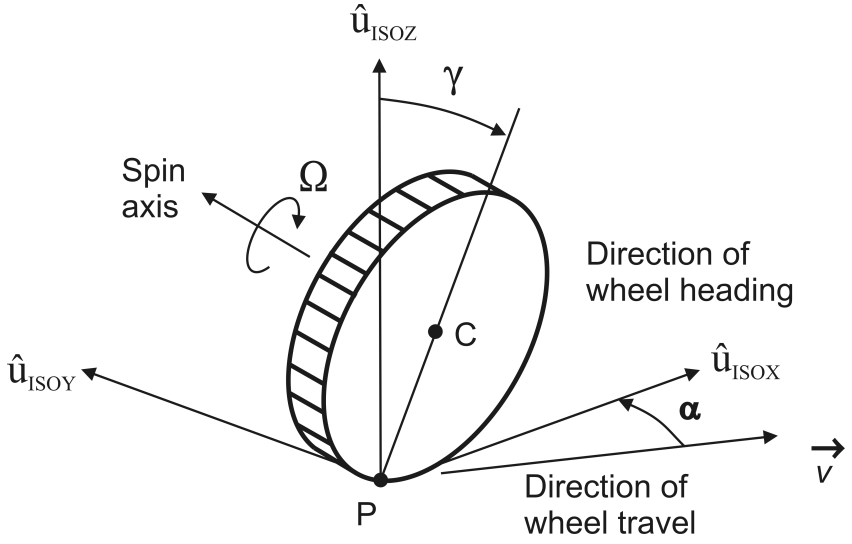


Figure 3.4: ISO Tire Axis System

The ISO system was developed for vehicles rolling over flat roads [1], but can be extended to vehicles traveling over 3-D trajectories. Expressions for each of the tire ISO unit vectors can be determined using simple vector operations. It is to be noted that the road profile is specified as a general function of x and y, i.e. $z = f(x, y)$. The ISO Z unit vector ($\hat{u}_{ISOZ}$) is in the same direction of the surface normal at the point of contact [1]. The surface normal (n_z) can be obtained using vector calculus, as shown in Figure (3.5), since the road surface is specified as a general function [92]. Thus, the ISO Z unit vector is calculated by normalizing the surface normal and evaluating it at the point of contact, as shown in Equation (3.4).

$$\{u_{ISOZ}\} = \frac{1}{\sqrt{\left(\frac{\partial z}{\partial x}\right)^2 + \left(\frac{\partial z}{\partial y}\right)^2 + 1}} \begin{Bmatrix} -\frac{\partial z}{\partial x} \\ -\frac{\partial z}{\partial y} \\ 1 \end{Bmatrix} \tag{3.4}$$

The ISO X unit vector ($\hat{u}_{ISOX}$) points in the direction of wheel heading and is perpendicular to both the ISO Z unit vector and the axis of symmetry of the tire ($\hat{u}_{SymAxis}$) [1], which is equivalent to the spin axis of the tire that is shown in Figure (3.4). The ISO X unit vector is determined using Equation (3.5).

$$\hat{u}_{ISOX} = \frac{\hat{u}_{SymAxis} \times \hat{u}_{ISOZ}}{|\hat{u}_{SymAxis} \times \hat{u}_{ISOZ}|} \tag{3.5}$$

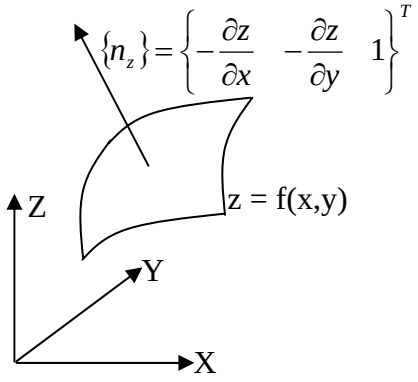


Figure 3.5: The Normal Vector of a Parametric Surface

The ISO Y unit vector ($\hat{u}_{ISOY}$) is perpendicular to both the ISO X and the ISO Z unit vectors and is chosen to form a right handed coordinate system [1]. The ISO Y unit vector is calculated using Equation (3.6).

$$\hat{u}_{ISOY} = \hat{u}_{ISOZ} \times \hat{u}_{ISOX} \tag{3.6}$$

The ISO Y unit vector will point in the lateral direction of the vehicle and thus the lateral force (F_y) and rolling resistance moment (M_y) act along this unit vector. The longitudinal force (F_x) and the overturning moment (M_x) act along the ISO X unit vector, and the normal force (F_z) and the aligning moment (M_z) act along the ISO Z unit vector.

3.2.5 Determine Expressions for the Most Common Kinematic Inputs Required by Tire Models

There are several different variables that are used to obtain the magnitudes of the tire forces and moments. The majority of tire models require a subset of seven parameters as input: the normal force, the inclination angle (camber angle), the forward speed of the tire, the spin rate, the effective rolling radius, the longitudinal slip, and the lateral slip angle [89]. These variables are dependent on the position, velocity and orientation of the tire with respect to the ground reference frame as well as the directions of the ISO unit vectors. The ground reference frame is the inertial reference frame.

The inclination angle (γ) is the angle between the ISO Z unit vector and the tire plane, as shown in Figure (3.4). The tire plane is the plane of the planar disk that represents the tire [76]. The inclination angle is a measure of how much the wheel is tilted away from the plane made by the X and Z ISO unit vectors. It is also referred to as the camber angle, but the term inclination angle will be used throughout this thesis. An expression for the inclination angle can be determined from geometric relations using the axis of symmetry of the tire and the ISO unit

vectors, and is shown in Equation (3.7).

$$\gamma = \arcsin((\hat{u}_{ISOY} \times \hat{u}_{SymAxis}) \cdot \hat{u}_{ISOX}) \tag{3.7}$$

The forward velocity of the tire (V_{XC}) is obtained by determining the component of the velocity of the tire center along the ISO X unit vector, and is shown in Equation (3.8). The velocity of the tire center ($\vec{V}_C$) is obtained from the vehicle model.

$$V_{XC} = \vec{V}_C \cdot \hat{u}_{ISOX} \tag{3.8}$$

The angular velocity of the tire ($\vec{\omega}_C$) is the vector sum of three components: the inclination rate ($\vec{\omega}_C \cdot \hat{u}_{ISOX}$), the spin rate ($\Omega$), and the yaw rate ($\omega_Z$) [8].

$$\vec{\omega}_C = (\vec{\omega}_C \cdot \hat{u}_{ISOX})\, \hat{u}_{ISOX} + \Omega \hat{u}_{SymAxis} + \omega_Z \hat{u}_{ISOZ} \tag{3.9}$$

The angular velocity components can be orthogonal, or non-orthogonal, and in fact are non-orthogonal for an inclined wheel ($\hat{u}_{SymAxis} \neq \hat{u}_{ISOY}$ and $\omega_Z \neq \vec{\omega}_C \cdot \hat{u}_{ISOZ}$). The spin rate is the important component since it is the angular speed of the tire about its symmetry axis as shown in Figure (3.4). An expression for the spin rate of the tire can be determined by rearranging Equation (3.9).

$$\Omega = \vec{\omega}_C \cdot \hat{u}_{SymAxis} - (\omega_Z \hat{u}_{ISOZ}) \cdot \hat{u}_{SymAxis} \tag{3.10}$$

Note, the ISO X unit vector will always be perpendicular to the axis of symmetry of the tire (refer to Equation (3.5)). Equation (3.10) can be simplified by substituting Equation (3.7) into it.

$$\Omega = \vec{\omega}_C \cdot \hat{u}_{SymAxis} - \omega_Z \sin(\gamma) \tag{3.11}$$

Equation (3.12) can be used to rearrange Equation (3.11) so that it is in terms of quantities that are easily measured or obtained during simulation. Equation (3.12) is determined from geometrical relations when the wheel is inclined.

$$\omega_Z = \vec{\omega}_C \cdot \hat{u}_{ISOZ} - \Omega \sin(\gamma) \tag{3.12}$$

The final form of the spin rate is shown in Equation (3.13).

$$\Omega = \frac{\vec{\omega}_C \cdot \hat{u}_{SymAxis} - (\vec{\omega}_C \cdot \hat{u}_{ISOZ}) \sin(\gamma)}{\cos^2(\gamma)} \tag{3.13}$$

The effective rolling radius (R_{eff}) is defined as the radius of an imaginary disk that rolls without slipping and has the same free rolling properties as the actual tire — it is the radius of the tire in the free rolling state. The free rolling state is defined as a loaded wheel that is not subjected to any driving or braking torques [1]. The effective rolling radius is closely related to the spin rate at zero slip angle and

zero inclination angle (Ω_0), and is shown in Equation (3.14). The slip angle will be defined later in this section.

$$R_{eff} = \frac{V_{XC}}{\Omega_0} \tag{3.14}$$

The forward velocity and Ω_0 are normally measured experimentally when the above formula is used [1]. The effective rolling radius will not be equal to the unloaded radius (R_u) or the loaded radius (R_l) of the tire since there is always some slip and distortion present in the contact patch of the tire. It usually lies between the unloaded radius and loaded radius as shown in Figure (3.6).

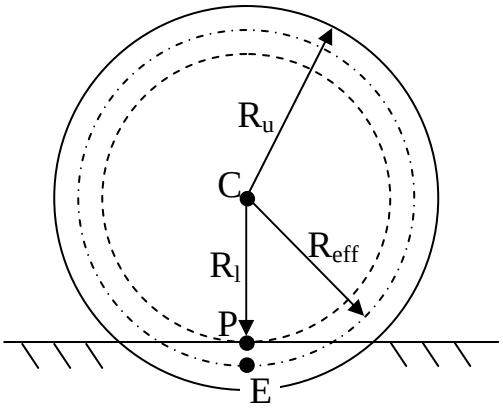


Figure 3.6: Different Measures of the Tire Radius

There are several different methods that can be used to estimate the effective rolling radius, and some of the most common methods are as follows.

- Use the loaded radius (R_l) — $R_{eff} = R_l = |\vec{r}_{P/C}|$
- Use the unloaded radius (R_u) — $R_{eff} = R_u$
- Use a formula created by the developers of MSC.ADAMS [7], and shown in Equation (3.16)

$$\rho = R_u - R_l \tag{3.15}$$

$$R_{eff} = R_u - \rho_{FZ0}\left(D\arctan\left(B\frac{\rho}{\rho_{FZ0}}\right) + F\frac{\rho}{\rho_{FZ0}}\right) \tag{3.16}$$

In Equation (3.16), ρ is the tire deflection, ρ_{FZ0} is the tire deflection at some nominal value of the normal force, and B, D and F are parameters that describe the tire [7]. The formula proposed by the developers of MSC.ADAMS was created based on experimental results. They found that the effective rolling radius is dependent on the normal force [74]. The method is normally used in conjunction with the Pacejka

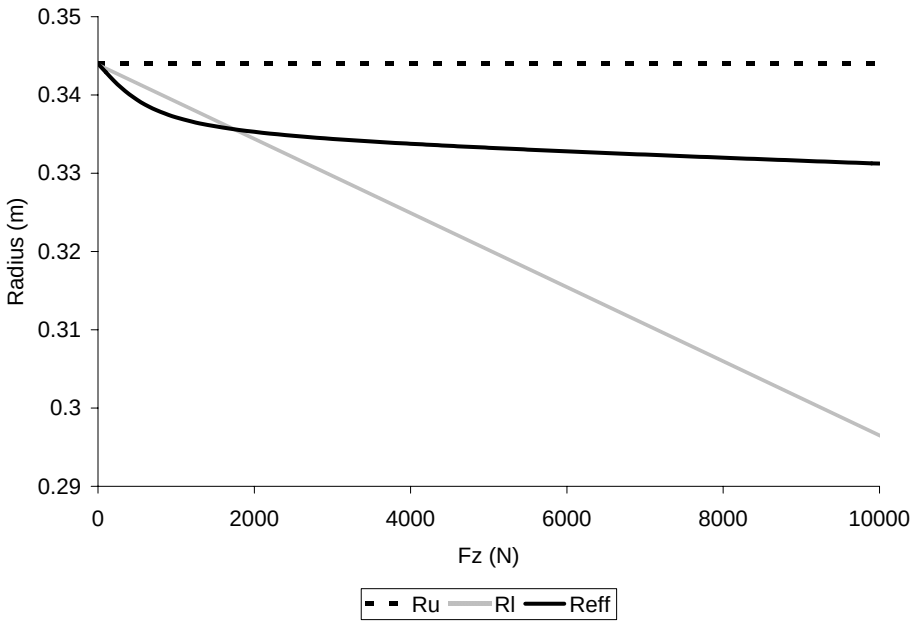


Figure 3.7: Estimates of the Effective Rolling Radius

tire model since this tire model can only be used if experiments are done on the tire. In this thesis, Equation (3.16) will be referred to as the ADAMS/Pacejka method for estimating the effective rolling radius.

Figure (3.7) compares the three methods for a typical sport utility vehicle tire and shows that at higher normal forces the effective rolling radius should be estimated using Equation (3.16). Note, Reff in Figure (3.7) was obtained using the ADAMS/Pacejka method and the parameters used to generate the figure can be found in Appendix (C). At lower normal forces the effective rolling radius can be estimated by assuming that it is equal to the loaded radius. It can be noted that the loaded radius is obtained once the location of the contact point is known. The method chosen to estimate the effective rolling radius is dependent on the desired accuracy.

The longitudinal slip (S) is a measure of how fast the wheel is spinning relative to its free rolling state. During braking, the wheel will spin slower than the free rolling case and the longitudinal slip is negative. The opposite is true during acceleration, and the longitudinal slip is positive. ISO defines the longitudinal slip by Equation (3.17) [1].

$$S = \frac{\Omega - \Omega_0}{|\Omega_0|} \tag{3.17}$$

Substituting Equation (3.14) into Equation (3.17) results in a popular definition of the longitudinal slip since the variables are easily measurable — V_{XC} is obtained directly from the vehicle model, and R_{eff} can be approximated using known vari-

ables.

$$S = \frac{\Omega R_{eff} - V_{XC}}{|V_{XC}|} \tag{3.18}$$

It can be noted that Ω_0 is defined for a wheel with zero inclination angle and thus Equation (3.17) and (3.18) are defined for a wheel with zero inclination angle.

Modifications have to be made to Equation (3.18) to calculate the longitudinal slip of an inclined wheel. The numerator of equation (3.18) gives the negative component of the velocity of point E.

$$S = \frac{-V_{XE}}{|V_{XC}|} \tag{3.19}$$

Point E is a distance of R_{eff} from the wheel center and is located along the vector $\vec{r}_{P/C}$, and is shown in Figure (3.6). It can be noted that point E is close to the contact point, but does not coincide with it unless $R_{eff} = R_l$. To be applicable to inclined wheels, Equation (3.19) must be written in terms of the velocity of point E only. In order to do this it is necessary to consider a wheel that translates, tilts and yaws with the actual tire, but does not rotate with it. This wheel is known as the geometrical wheel [76]. A g will be used to denote a quantity that measures a characteristic on the geometrical wheel. Therefore, Equation (3.19) can be written in terms of the velocity of point E.

$$S = \frac{-V_{XE}}{|V^g_{XE}|} = \frac{-\left(\left(\vec{V}_C + \vec{\omega}_C \times \vec{r}_{E/C}\right) \cdot \hat{u}_{ISOX}\right)}{\left|\left(\vec{V}_C + (\vec{\omega}_C - \Omega\hat{u}_{SymAxis}) \times \vec{r}_{E/C}\right) \cdot \hat{u}_{ISOX}\right|} \tag{3.20}$$

It can be noted that, $V_{XC} = V^g_{XE}$ for a free rolling wheel with no inclination angle. It can also be noted that the velocity of point E on the geometrical wheel is calculated in a way that permits the tire to be inclined with respect to its rotational axis; refer to Figure (3.8). In Equation (3.20), the spin rate is subtracted from the angular

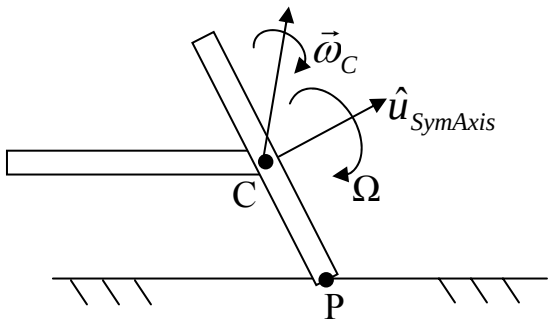


Figure 3.8: Inclined Wheel with Respect to the Rotational Axis

velocity to obtain the velocity of point E on the geometrical wheel, and is thus

valid for inclined and non-inclined wheels. It can be further noted that all of the equations previously derived (Equations (3.7)-(3.13)) also allow for a wheel to be inclined with respect to its rotational axis.

The longitudinal slip and the longitudinal force (F_x) should always oppose the sign of V_{XE}, and this is the reason why the absolute value is used in Equations (3.17)-(3.20). Equation (3.20) is recommended by Blundell and Harty [23] and by Pacejka [74], and will be used throughout this thesis to calculate the longitudinal slip.

The lateral slip angle (α) is a measure of slip in the lateral direction and exists when the velocity of the tire is not completely along the ISO X axis. It is defined as the angle between the direction of wheel heading and the direction of wheel travel and is shown in Figure (3.4) [1]. The ISO definition of the lateral slip angle is given by Equation (3.21) [1].

$$\alpha = \arctan\left(\frac{V_{YP}}{|V_{XP}|}\right) \tag{3.21}$$

The velocity of point P is used to allow the slip angle to be calculated for a wheel with an inclination angle, and because it is at this point where the tire slip occurs. This is similar to using point E instead of the wheel center for the longitudinal slip. The contact point is not a fixed point on the tire and wheel spin motions must be ignored to follow its trajectory. Thus, the expression for the lateral slip angle should be in terms of the velocity of point P on the geometrical wheel.

$$\begin{aligned} \alpha &= \arctan\left(\frac{V^g_{YP}}{|V^g_{XP}|}\right) \\ &= \arctan\left(\frac{\left(\vec{V}_C + (\vec{\omega}_C - \Omega\hat{u}_{SymAxis}) \times \vec{r}_{P/C}\right) \cdot \hat{u}_{ISOY}}{\left|\left(\vec{V}_C + (\vec{\omega}_C - \Omega\hat{u}_{SymAxis}) \times \vec{r}_{P/C}\right) \cdot \hat{u}_{ISOX}\right|}\right) \end{aligned} \tag{3.22}$$

The absolute value is used to ensure that the value of the lateral slip angle is correct. It should be positive when V_{YP} is positive so that the lateral force (F_y) is positive. This allows its value to be correct whether the vehicle is going forward or in reverse. The ISO definition will be adopted in this thesis.

The tire forces do not develop instantly, but build up as the tire rolls. A dynamic lag can be introduced to model this characteristic and there are two methods that are commonly used to model the lag and these are as follows.

- Use a steady state tire model that takes delayed longitudinal slip and lateral slip angle as input
- Use a tire model that has the dynamics built into it

The preferred method is the first approach since it permits the analyst to choose a desired tire model independent of the method used to introduce dynamic lag [75, 81].

The accepted behavior to model dynamic lag is to model the delay as two first order differential equations, one for the longitudinal slip, and one for the lateral slip angle, with a parameter called the relaxation length. The relaxation length is similar to a time constant except that it has units of length instead of time [22, 74, 81]. Two differential equations have been proposed by Bernard and Clover [22] and modified by Pacejka [74] to model the dynamic lag and are shown in Equations (3.23) and (3.24).

$$\frac{dS}{dt} = \frac{-V_{XE}}{B_{long}} - \frac{S|V_{XE}^g|}{B_{long}} \tag{3.23}$$

$$\frac{d(\tan(\alpha))}{dt} = \frac{V_{YP}^g}{B_{lat}} - \frac{\tan(\alpha)|V_{XP}^g|}{B_{lat}} \tag{3.24}$$

B_{long} and B_{lat} are the relaxation lengths in the longitudinal and lateral directions respectively. It is important to note that the steady state response of the two differential equations is the kinematic slip equations previously derived. It can also be noted that the kinematic slip equations can not be used when the velocity is zero since this quantity appears in the denominator of the equations. However, the delayed slip equations can be used when the velocity is equal to zero. This property is important for driving simulators since in a driving simulator the vehicle may come to a complete stop [22].

When the differential equations were derived by Bernard and Clover, they assumed constant values for the relaxation lengths. However, Pacejka [74] did some physical tests on the tire transients and determined that the relaxation lengths can vary significantly with normal force (F_Z), longitudinal slip (S), and lateral slip angle (α). The developers of MSC.ADAMS created the stretched string method to model the dependency of the relaxation length on the tire normal force and inclination angle (γ) [7]. They developed this approach by considering the behavior of a string that follows the tire circumference and is kept under a certain pretension by a uniform radial force distribution that is comparable to the inflation pressure of real tires [74]. The stretched string equations are shown in Equations (3.25-3.27).

$$dF_Z = \frac{F_Z - F_{Z0}}{F_{Z0}} \tag{3.25}$$

$$B_{long} = LS_k R_u \frac{F_Z}{F_{Z0}} \left(PT_{X1} + PT_{X2} dF_Z\right) e^{PT_{X3} dF_Z} \tag{3.26}$$

$$B_{lat} = LS_k PT_{Y1} F_{Z0} \sin\left(2 \arctan\left(\frac{F_Z}{PT_{Y2} F_{Z0} LS_\alpha}\right)\right) (1 - PK_{Y3}|\gamma|) R_u LS_\alpha \tag{3.27}$$

where, F_{Z0} is the nominal normal force between the tire and the road, γ is the inclination angle, R_u is the unloaded radius, and the rest of the parameters are determined from physical tests.

Pacejka [74] also determined that the accuracy of the delayed slip equations is compromised at low speeds since for small values of V_{XP}^g and V_{XE}^g, Equations (3.23) and (3.24) act as integrators of V_{YP}^g and V_{XE}. This can give rise to large values of longitudinal slip and lateral slip angle. To eliminate S and α from taking on

unreasonable values, Pacejka [74] proposed setting each of the equations equal to zero when the forward speed was below a threshold and when the longitudinal slip and lateral slip angle were over a certain threshold and growing. The algorithm suggested by Pacejka [74] is shown in Table (3.1).

if:	$\lvert V_{XE}^g \rvert < V_{low}$ and $\lvert S \rvert > S_{high}$ and $\left(\frac{-V_{XE}}{B_{long}} - \frac{S\lvert V_{XE}^g \rvert}{B_{long}} \right) S > 0$
then:	$\frac{dS}{dt} = 0$
else:	use Equation (3.23)
if:	$\lvert V_{XP}^g \rvert < V_{low}$ and $\lvert \alpha \rvert > \alpha_{high}$ and $\left(\frac{V_{YP}^g}{B_{lat}} - \frac{\tan(\alpha)\lvert V_{XP}^g \rvert}{B_{lat}} \right) \alpha > 0$
then:	$\frac{d(\tan(\alpha))}{dt} = 0$
else:	use Equation (3.24)

Table 3.1: Pacejka's Algorithm for Limiting the Derivatives of the Tire Slip Parameters When Tire Transients are Important

If it is anticipated that the vehicle will come to a stop during simulation and start up again, or if tire transients are important, then the delayed slip equations should be used. These equations are appended to the governing equations so that they are solved at each time step to obtain the slip parameters. The algorithm suggested by Pacejka will be used when results from physical tests are available; else constant values will be used for the relaxation lengths.

3.2.6 Use a Tire Model to Determine the Remaining Tire Forces and Moments

The magnitude of the tire forces and moments are obtained using a tire model given the kinematic variables as input. These forces are then acted along the direction of the ISO unit vectors. The tire model used to calculate the tire forces and moments in this thesis is a mathematics-based tire model. As previously mentioned, these types of tire models are designed to match the results obtained from doing physical tests on the tire and are used to analyze the vehicle in acceleration, braking, and cornering maneuvers.

There are a few observations that can be made about pneumatic tires in regards to the tire force and moment generation and some of them are presented below [23, 74, 68]. The tire models attempt to capture the behavior explained below and thus these observations should be seen in the the tire curves obtained from each tire model.

- The lateral (F_y) and longitudinal (F_x) forces increase with increasing normal force, but the relationship is non-linear

- When the slip parameters are low, F_x is proportional to the longitudinal slip (S), and F_y is proportional to the lateral slip angle (α)
- When the slip parameters are high, F_x and F_y become saturated and their maximum values are determined by the coefficient of friction between the tire and the road
- When both F_x and F_y are present (combined slip situation), the value of F_x will be lower than it would have been if there was a case of pure longitudinal slip, and the same is true for F_y but with the case of pure lateral slip
- The inclination angle decreases the size of the contact patch and thus has a diminishing effect on the forces and moments generated; however, a small inclination angle can lead to an increase in the lateral force
- An aligning moment (M_z) is present when the tire is steering and tends to bring the tire back to the straight ahead position, and depends primarily on the slip angle and secondarily on the normal force, longitudinal slip, and inclination angle

There are several different tire models that exist and a review of some of them is described by Pacejka and Sharp [75], and by Blundell and Harty [23]. The different tire models can vary greatly and their complexity is dependent on the number of parameters that are used to construct the tire curves from physical tests. Some of the more complex tire models require sophisticated tests to be done on the tire to obtain the parameters that are used to construct the tire curves. Also, the more sophisticated the tire model, the more computationally expensive it is. The desired tire model is not necessarily the most sophisticated tire model, but the one that gives the desired level of accuracy for the vehicle test being performed. Two of the most common tire models will be used in this thesis, and each of them will be described in the next two sections.

Fiala Tire Model

The Fiala tire model, developed by Fiala in 1954 [23, 38] is a simplified model that does not require extensive and complicated physical tests to determine the necessary parameters. It is the default tire model in MSC.ADAMS and is constructed using 6 basic parameters. This simplified tire model has several limitations and these are as follows.

- The model does not consider changes in the inclination angle
- The tire curves must pass through the origin (the forces and moments are zero when the slip parameters are zero)
- The model does not represent the combined slip situations accurately

- The lateral and longitudinal stiffness are constant and thus do not change with normal load
- The overturning moment (M_x) is assumed to be zero

Despite these limitations, the Fiala tire model is still widely used because of its simplicity. The model requires the normal force (F_z) and a subset of the kinematic variables (S, Ω, α) as input. The equations for the Fiala tire model are presented in Equations (3.28)-(3.35).

$$S_{L\alpha} = \sqrt{S^2 + \tan^2(\alpha)} \tag{3.28}$$

$$\mu = \mu_0 - S_{L\alpha}(\mu_0 - \mu_1) \tag{3.29}$$

$$H = 1 - \frac{C_\alpha |\tan(\alpha)|}{3\mu |F_z|} \tag{3.30}$$

$$F_x = \begin{cases} C_S S & \text{if } |S| < \left|\frac{\mu F_z}{2C_S}\right| \\ \text{sign}(S)\left(\mu F_z - \left(\frac{(\mu F_z)^2}{4|S|C_S}\right)\right) & \text{otherwise} \end{cases} \tag{3.31}$$

$$F_y = \begin{cases} -\mu |F_z| (1 - H^3)\text{sign}(\alpha) & \text{if } |\alpha| < \arctan\left(\left|\frac{3\mu F_z}{C_\alpha}\right|\right) \\ -\mu |F_z| \text{sign}(\alpha) & \text{otherwise} \end{cases} \tag{3.32}$$

$$M_x = 0 \tag{3.33}$$

$$M_y = \begin{cases} C_r F_z & \text{if } \Omega < 0 \\ -C_r F_z & \text{otherwise} \end{cases} \tag{3.34}$$

$$M_z = \begin{cases} \mu F_z D_2 (1 - H) H^3 \text{sign}(\alpha) & \text{if } |\alpha| < \arctan\left(\left|\frac{3\mu F_z}{C_\alpha}\right|\right) \\ 0 & \text{otherwise} \end{cases} \tag{3.35}$$

The list of the parameters used in the Fiala tire model is shown in Table (3.2).

The cornering stiffness, lateral stiffness, rolling resistance moment coefficient, peak coefficient of friction, sliding coefficient of friction, and the width of the tire are the 6 parameters that are required as input to the Fiala tire model. The stiffness coefficients are measured from test results or can be estimated using an approximation such as the method proposed by Hewson [50]. The comprehensive slip ratio is the resultant of the longitudinal slip and lateral slip angle and determines the current slip state of the tire. It is used to estimate the current coefficient of friction between the tire and the ground which is somewhere between the static coefficient of friction and the sliding coefficient of friction. The current coefficient of friction is estimated using linear interpolation.

Tire curves from the Fiala tire model are shown in Figures (3.10) and (3.11). The figures also compare the results obtained from this simplified model to those obtained from the 2002 Pacejka model, a more sophisticated tire model. General observations from the figures will be made in the next section.

The Fiala tire model has several if statements that are used to construct its equations, and thus numerical values need to be sent to the model to obtain the

μ	The current value of the coefficient of friction
μ_0	The peak coefficient of friction between the tire and the ground (the static coefficient of friction)
μ_1	The sliding coefficient of friction between the tire and the ground (occurs at pure sliding; slip=100%)
C_S	The longitudinal tire stiffness (the slope of the F_x vs S tire curve at $S = 0$ (units - force))
C_α	The lateral tire stiffness (the slope of the F_y vs α tire curve at $\alpha = 0$ (units - force/radian))
C_r	The rolling resistance moment coefficient (units - length)
D_2	The tire width (units - length)
$S_{L\alpha}$	The comprehensive slip ratio
H	The slip angle intermediate parameter

Table 3.2: The List of Parameters Used in the Fiala Tire Model Equations

tire forces and moments. This implies that there is no closed form solution. Thus, in the symbolic formulation stage there is no possibility of obtaining a closed form solution to the governing equations that includes the tire forces and moments being calculated using the Fiala tire model. This is also the case when other tire models are adopted. Therefore, in a multibody systems context, the tire forces and moments need to be evaluated during simulation and not during formulation. During simulation, the numeric values for the tire model inputs will be known. Thus, there is a need to develop a simulation code structure that permits the governing equations to be generated using a symbolic approach and permits the tire forces and moments to be evaluated during simulation. A simulation code structure has been created to do this, and details can be found in Chapter (4).

Pacejka Tire Model

The Pacejka tire model is known as one of the most accurate tire models and is widely used and accepted for use in handling studies of the vehicle on flat roads [23]. The Pacejka tire model is also known as the magic tire formula and has been developed based on previous versions of the model. The magic formula tire models were developed using trigonometric expressions to match the results obtained from physical tests [23]. These types of tire models are always going through changes as different researchers propose new versions of the model. The first version was introduced by Bakker et al. in 1986 [18], and was very simple and only had expressions for the longitudinal force, the lateral force, and the overturning moment. The longitudinal force was a function of only the longitudinal slip and the lateral force and aligning moment were a function of only the lateral slip angle. However, the first version outlined the general form of the tire curves [18], and the most current versions are still based on the general form. One of the most recent versions is the 2002 Pacejka tire model. This tire model takes into consideration several of the

non-linear effects, including the inclination angle and combined slip (the friction circle) [74].

The general form of the magic formula model that is used to calculate the lateral force, longitudinal force and aligning moment is shown below:

$$x = X + S_h \tag{3.36}$$
$$y(x) = D\sin\left(C\arctan\left(Bx - E\left(Bx - \arctan\left(Bx\right)\right)\right)\right) \tag{3.37}$$
$$Y(X) = y(x) + S_v \tag{3.38}$$

where, Y (and y) is the force or moment in consideration, X (and x) is the slip parameter, S_h is the horizontal shift, S_v is the vertical shift, and A through E are the magic formula coefficients. The horizontal and vertical shifts are used to account for the effects of ply steer (the tire tends to drift one way), conicity (the tire is not perfectly circular), and inclination angle.

The magic formula coefficients are best described with reference to Figure (3.9). The peak value of the tire curve is governed by the peak factor (D). The stretching

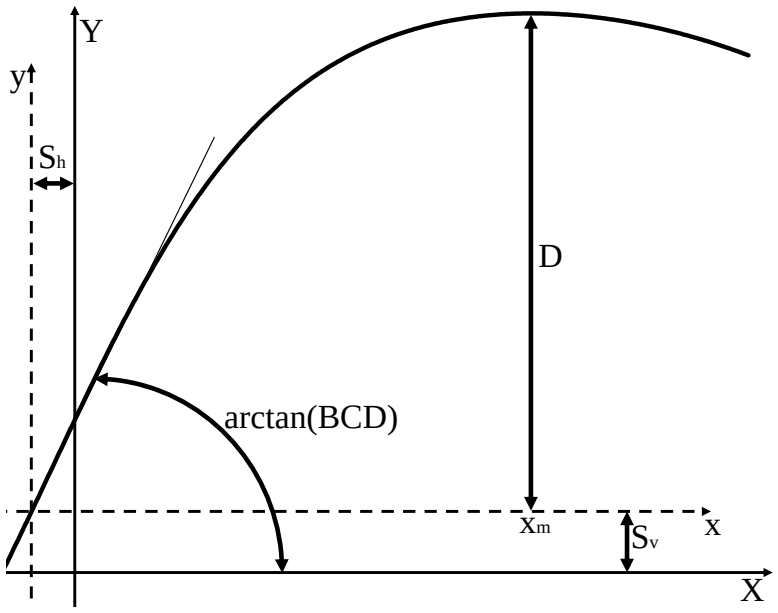


Figure 3.9: Pacejka Tire Model Coefficients

of the tire curve in the x direction is controlled by the shape factor (C). The stiffness factor (B) controls the slope of the tire curve at the origin. However, the slope of the tire curve is decided by B, C, and D. The non-linearity of the tire curve is governed by the curvature factor (E), which also affects the position where the peak coefficient occurs [23].

A magic tire formula is constructed by determining expressions for each of the factors for each of the force and moment curves given the normal force and kinematic

parameters as input. These factors are usually dependent on several coefficients that are determined from physical tests performed on the tire. One of the most basic magic formula tire models is shown below for the lateral force and is dependent on only eight coefficients [74].

$$D = \left(k_0 F_z + k_1 F_z^2\right)\left(1 - k_2 \gamma^2\right) \tag{3.39}$$

$$BCD = k_3 \sin\left(2 \arctan\left(\frac{F_z}{k_4}\right)\right) \tag{3.40}$$

$$B = \frac{BCD}{CD} \tag{3.41}$$

$$E = \left(k_5 F_z + k_6\right)\left(1 - k_7 \text{sign}\left(\alpha\right)\gamma\right) \tag{3.42}$$

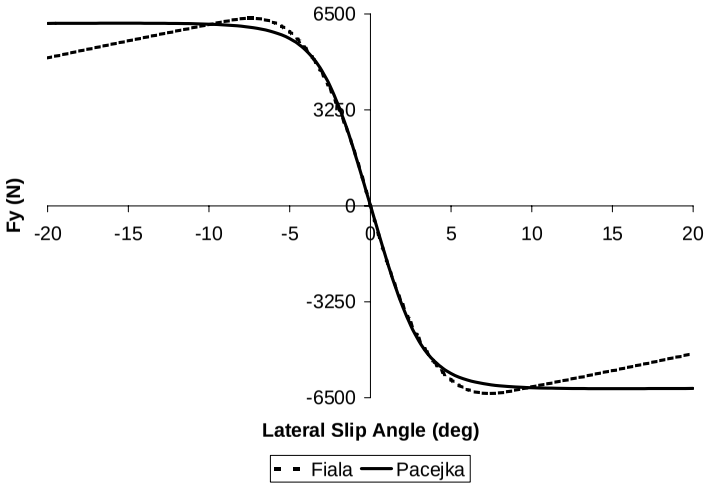
where,

k_0	The linear coefficient of friction
k_1	The coefficient of friction reduction (reduces μ with load)
k_2	The camber coefficient (reduces tire grip with camber)
k_3	The maximum cornering coefficient
k_4	The vertical load at which maximum cornering stiffness occurs
k_5, k_6, and k_7	The curvature factor coefficients

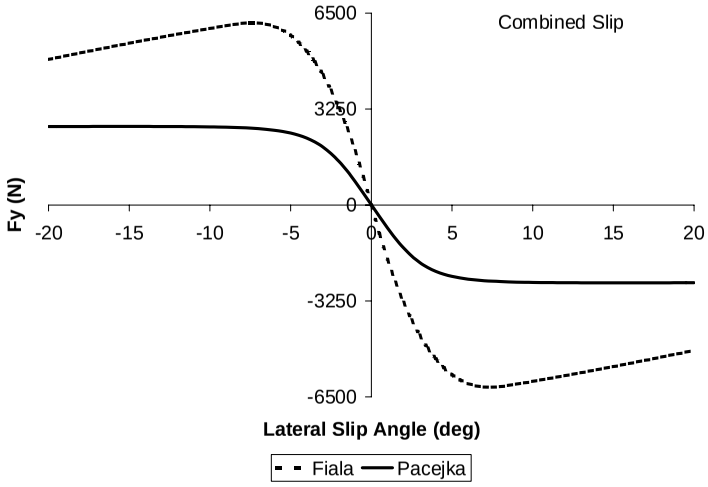
Table 3.3: The List of Coefficients Used in the Pacejka Tire Model Equations

Note, the shape factor is usually around 1.3 for passenger vehicles and is treated as a constant [74]. The complexity of the magic tire model can improve as the accuracy of each of the factors is increased. The 2002 Pacejka tire model for instance uses 117 parameters to describe a single tire [74]. It is to be noted that this model also includes expressions for the overturning moment and the rolling resistance moment. The equations for the 2002 Pacejka model are too lengthly and thus will not be presented here, but can be found in the book published by Pacejka [74].

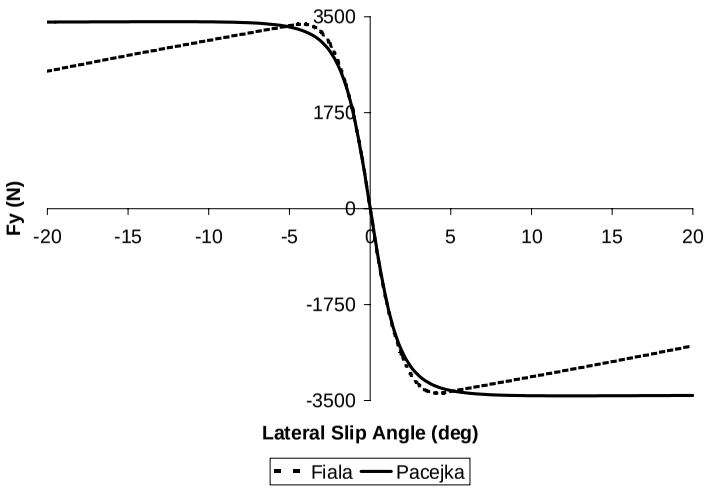
The 2002 Pacejka tire model will be used throughout this thesis and an example of the tire curves obtained from this model is presented in Figures (3.10) and (3.11). The tire curves obtained from the Fiala tire model are also shown in the figures to allow for a comparison between the two tire models. The parameters used to construct the tire curves are presented in Appendix (B). The curves presented in the figures represent the response of a typical tire for a generic four-wheeled vehicle. The nominal condition is defined as the state when the tire is acting at its nominal load with zero slip parameters and inclination angle ($S = \alpha = \gamma = 0$). The nominal load of the tire used for demonstration purposes is 5900N. The lateral force tire curve is used to show how the tire models respond to cases of combined slip, non-zero inclination angle, and changes in the normal load. An example of the tire curves for the other responses is shown for the nominal case only.



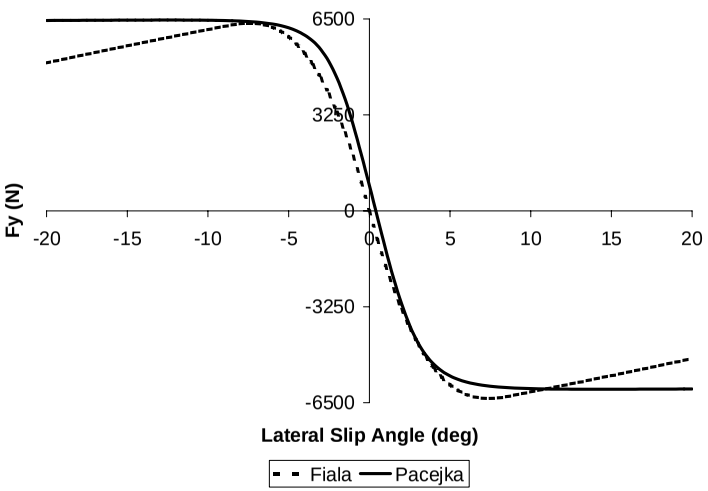
(a) Nominal Conditions



(b) $S = 0.1$

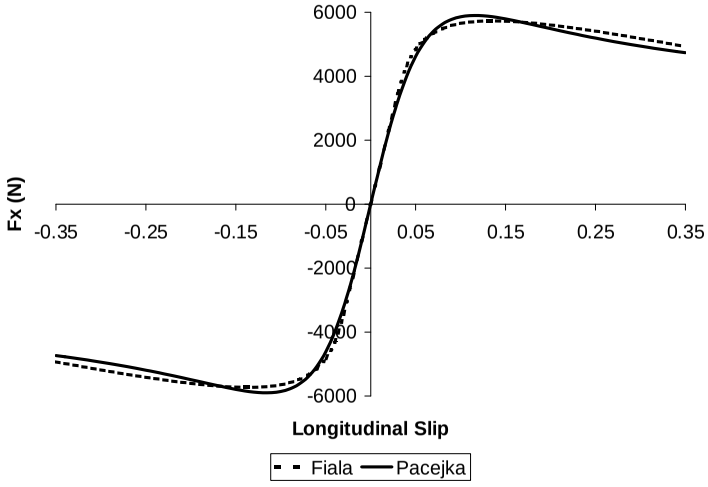


(c) $F_z = 2950N = \frac{1}{2}F_{znominal}$

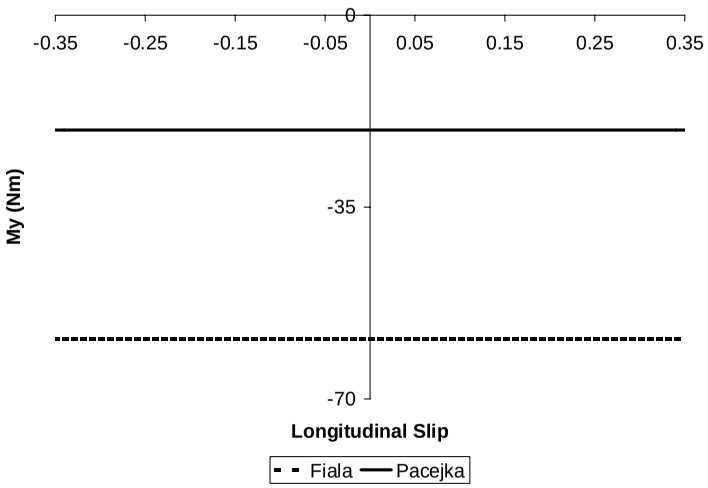


(d) $\gamma = -10deg.$

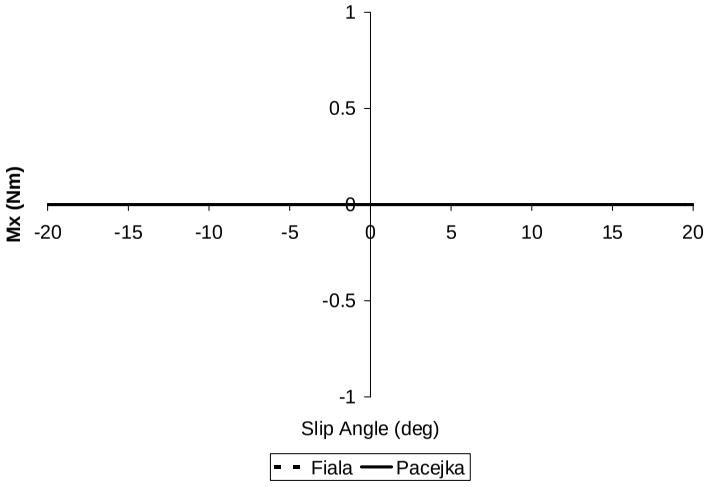
Figure 3.10: Comparison Between the Pacjeka 2002 and Fiala Tire Models in Different Scenarios for the Lateral Force vs Lateral Slip Angle



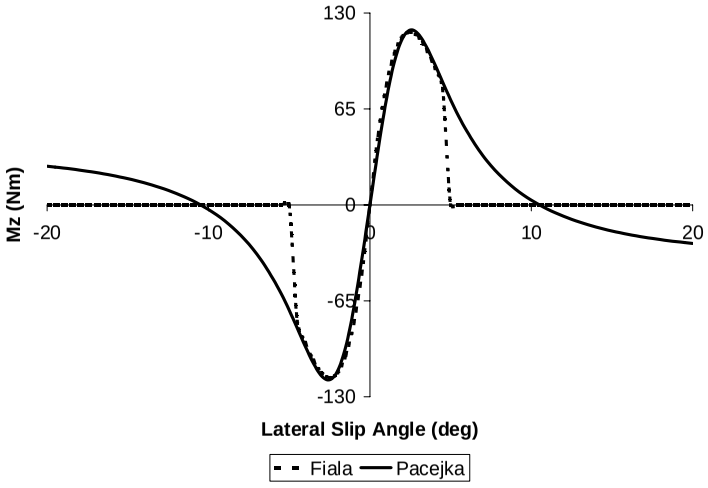
(a) F_x vs S



(b) M_y vs S



(c) M_x vs α



(d) M_z vs α

Figure 3.11: Comparison Between the Pacjeka 2002 and Fiala Tire Models for the Nominal Condition for the Remaining Tire Forces and Moments

The Fiala tire model and the 2002 Pacejka tire model give approximately the same result for the nominal case and changes in the normal load for pure lateral (or longitudinal) slip. For cases of combined slip or when an inclination angle is present, the Fiala tire model does not produce the same result as the Pacajka model, and the difference is more drastic in the combined slip situation, e.g. Figure (3.10 b). These two observations can also be made to the other tire curves when they are obtained for cases of combined slip and changes in the inclination angle. It can be seen from Figure (3.11) that the overturning moment is zero for all slip. In fact, most tire models will yield this result. It is to be noted that if there is an overturning moment, the 2002 Pacejka model will predict this effect, and it is zero for this particular tire only because of the coefficients that were obtained to represent the tire. The rolling resistance moment is a constant and independent of the slip parameters for both of the models. The 2002 Pacejka tire model does predict the rolling resistance moment as a function of the longitudinal velocity and the lateral force. It is just a constant in this case because of the coefficients that were used to represent the tire. The 2002 Pacejka model is more accurate than the Fiala tire model, but is more computationally expensive. The Fiala tire model is not as accurate but would be sufficient when the vehicle is not accelerating/braking and cornering at the same time (combined slip situation).

3.3 Calculation of the Contact Point Between the Tire and the Road

3.3.1 Comparison of Two Models

There are a number of approaches that can be used to calculate the contact point between the tire and the ground, and some of the most popular ones are as follows: the 3D equivalent volume model [3], the torus tire model [23], finite element methods (FEM) [73], and the thin disk tire model with variable radius [76]. Table (3.4) briefly explains how the tire and the road are modeled for each approach along with how the contact point is determined.

It is essential that the method used to calculate the contact point between the tire and the ground be amenable to being incorporated in a symbolic formulation approach of the governing equations. It is reminded that the tire model will be used for handling studies of the vehicle and not for durability studies. Thus, it was decided not to use finite element methods since they are computationally expensive, designed for both handling and durability studies of the vehicle, and are difficult to implement in a symbolic formulation of the governing equations. Furthermore, the torus tire model was eliminated because it requires more calculations to determine the contact point than the remaining two approaches. The 3D equivalent volume model is not easily implemented in a symbolic formulation approach.

Approach	Tire model	Road model	Calculation of the contact point
3D equivalent volume model	Modeled as a set of cylinders in the lateral direction along the circumference of the tire OR as a cylinder with a radius and width of the undeformed tire	Modeled as a series of triangular patches (triangular mesh) OR as a piecewise cubic spline	The tire penetration is determined by finding the volume of penetration between the undeformed tire and the road and comparing it to the case when the tire is rolling on a flat road. The contact point is determined by passing a line from the centroid of the penetration volume to the wheel center and the contact point is the point of intersection between this line and the ground. Refer to [11] for further details.
Torus tire model	The tire is modeled as a torus with radius equal to the radius of the undeformed tire and with the diameter of the carcass equal to the width of the undeformed tire	Same as previous approach	The contact point is determined by passing a line from the center of the undeformed tire carcass to a point on the road surface where the line is normal to the road surface. The point of intersection between this line and the road surface is the contact point. The tire penetration is the distance of intersection along this line between the undeformed tire carcass and the road surface. Refer to [23] for further information.
Finite Element Methods (FEM)	FEM mesh	FEM mesh	The contact point is determined by calculating the center of pressure of the contact area between the tire and the ground. The penetration of the tire is determined by calculating the deflection of each of the nodes in the FEM mesh of the tire. Refer to [73] for further details.
Thin disk tire model with variable radius	Modeled as a planar disk with variable radius	Modeled as an elevation function ($z = f(x, y)$)	The radius of the planar disk is varied along with its orientation in the wheel plane until the tire comes into contact at a single point with the ground (refer to Figure (3.2)). The tire penetration is the difference between the undeformed radius and the magnitude of the variable radius.

Table 3.4: Comparison Between the Different Approaches to Calculate the Contact Point Between the Tire and the Ground

However, by changing the road model to be an elevation function instead of a triangular mesh and by assuming that the tire can be represented as a cylinder, it can be easily implemented in a symbolic approach. Therefore, a simple model was created to compare the 3D equivalent volume model to the thin disk tire model with variable radius.

The simple model is the moving quarter car model. The quarter car model is a model that is typically used to test the durability of a single suspension system [100]. It is also used to determine the wheel hop frequency and the body motion frequency of the suspension/vehicle [42]. The suspension system is estimated as a lumped 2 DOF model with the suspension being represented by a single spring and damper and the tire by a single spring, and is shown in Figure (3.12). The input to this

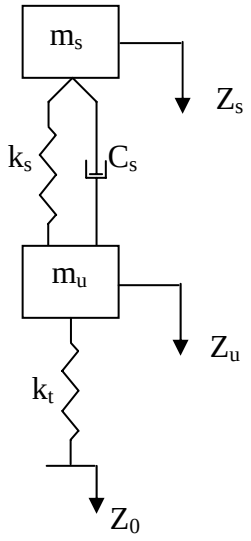


Figure 3.12: Original Quarter Car Model

model is typically a frequency input at the road (through Z_0). The typical model can be combined with either one of the approaches by calculating the contact point between the tire and the ground and using this information to calculate the normal force ($F_{\text{tire\_model}}$) that acts on the unsprung mass (m_u) of the suspension system, as is shown in Figure (3.13). With this particular model, the tire is modeled as a force input and not as a single spring. Instead of having the frequency as an input, the tire normal force is used as an input along with the velocity in the forward direction of the suspension system.

The 3D equivalent volume approach uses the concepts developed by Davis [32] to calculate the tire penetration and the location of the contact point. Davis developed the radial spring tire model, where the tire is modeled as a set of springs in the radial direction and the deflection of each of these spring determines the deformed area and the tire deflection [32]. This concept will be used here, but instead of modelling the tire as a set of springs, it will be modeled as a planar disk. Thus, the width of the tire is neglected and changes in the lateral direction are assumed

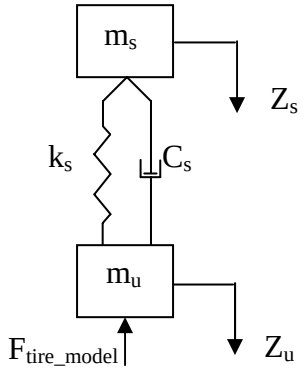


Figure 3.13: Moving Quarter Car Model

to have no effect in the location of the contact point. This is a reasonable estimate when the tire is used to model the vehicle in handling situations with small camber angles [98, 79, 33]. The tire deflection (δ) is determined by calculating the deformed area (A) between the undeformed tire (planar disk) and the ground and equating this to a deformed area that would be obtained if the tire was rolling on a flat surface, refer to Figure (3.14). The contact point is calculated by determining the

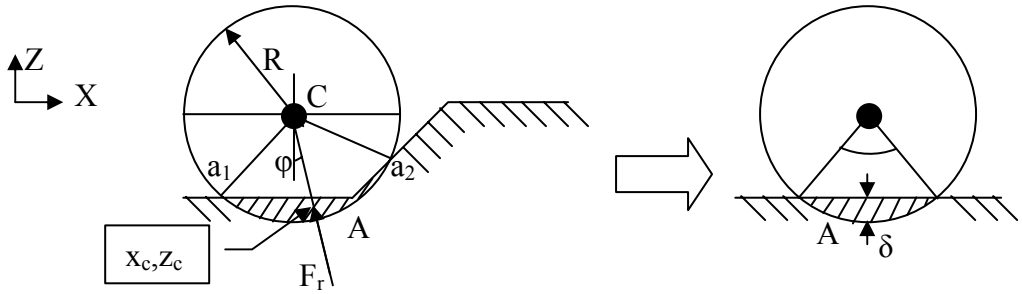


Figure 3.14: Calculation of the Tire Penetration Using the 3D Equivalent Volume Model

centroid of the deformed area (x_c, z_c) and creating a line that passes through the centroid and the wheel center, refer to Figure (3.15). The contact point is the intersection of this line with the road surface. The force (F_r) acts along this line at the road surface (the contact point) and the vertical component of this force ($F_{\text{tire\_model}}$) is what acts on the suspension system. Numerical integration is used to determine the area of penetration and the centroid of the area [93].

The thin disk tire model with variable radius was described in Section (3.2.2), and is discussed in detail in Section (3.3.2).

The two approaches were compared by having the moving quarter car model go over a 5cm high speed bump modeled as a sinusoidal function that is shown in

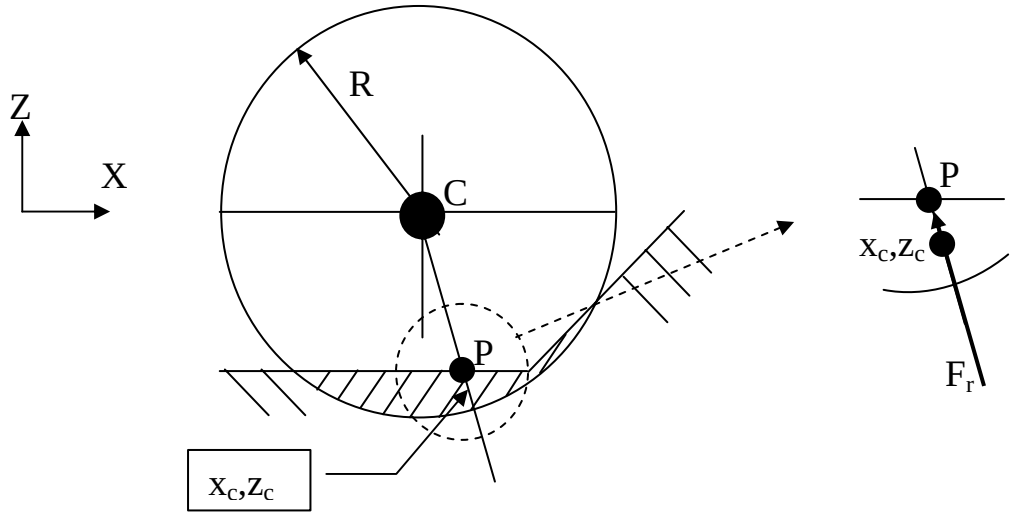


Figure 3.15: Determination of the Contact Point Using the 3D Equivalent Volume Model

Equation (3.43) and in Figure (3.16).

$$\text{Ground Profile} = \begin{cases} & x & < 0.5 \text{ m}: & 0 \text{ m} \\ 0.5 \text{ m} \leq & x & < 1 \text{ m}: & 0.025\left(1 - \cos\left(\left(\frac{2\pi}{0.5}\right)x\right)\right) \text{ m} \\ & x & \geq 1 \text{ m}: & 0 \text{ m} \end{cases} \quad (3.43)$$

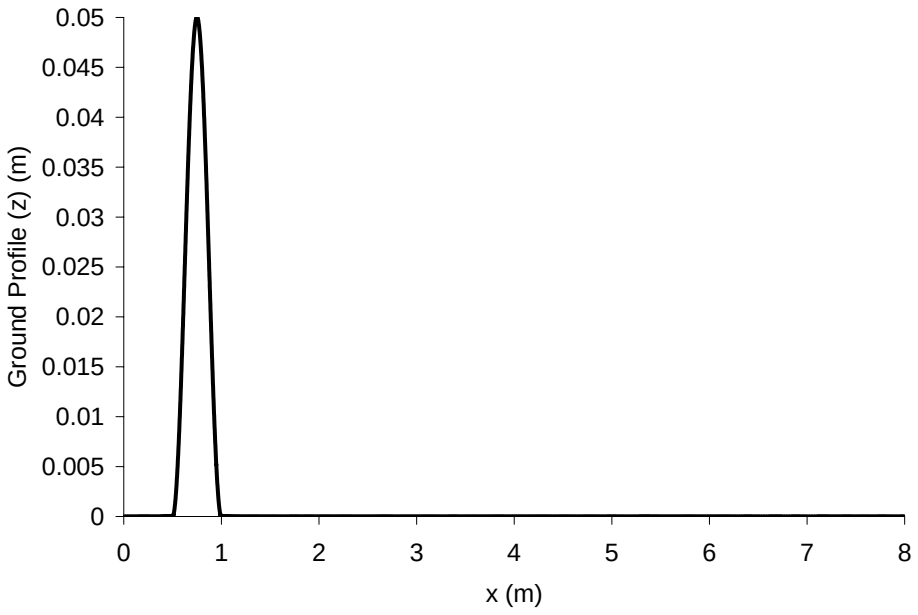


Figure 3.16: The Ground Profile (Speed Bump) Used to Compare the Two Approaches to Calculate the Contact Point

The suspension used for the comparison was the front left suspension of the Chevrolet Equinox. It was assumed to have an initial velocity of 2m/s. The parameters used for the analysis and the set of initial conditions can be found in Appendix (C). The equations representing this system were derived by hand and hard-coded into Matlab, and the analyzes were performed in Matlab using the ode45 solver with a tolerance of 1E-3 (both relative and absolute). Figures (3.17) and (3.18) show comparisons between results obtained from the two approaches and the following conclusions can be made.

- The thin disk tire model with variable radius gives a slightly larger result for both of the results obtained (Z_u, Z_s)
- Both of the results cross the x-axis at the same time (frequencies match)
- The two responses resemble one another
- The maximum value obtained from the thin disk tire model occurs slightly before the maximum value obtained from the 3D equivalent volume model approach
- The 3D equivalent volume model determines contact with the bump before the thin disk tire model with variable radius does

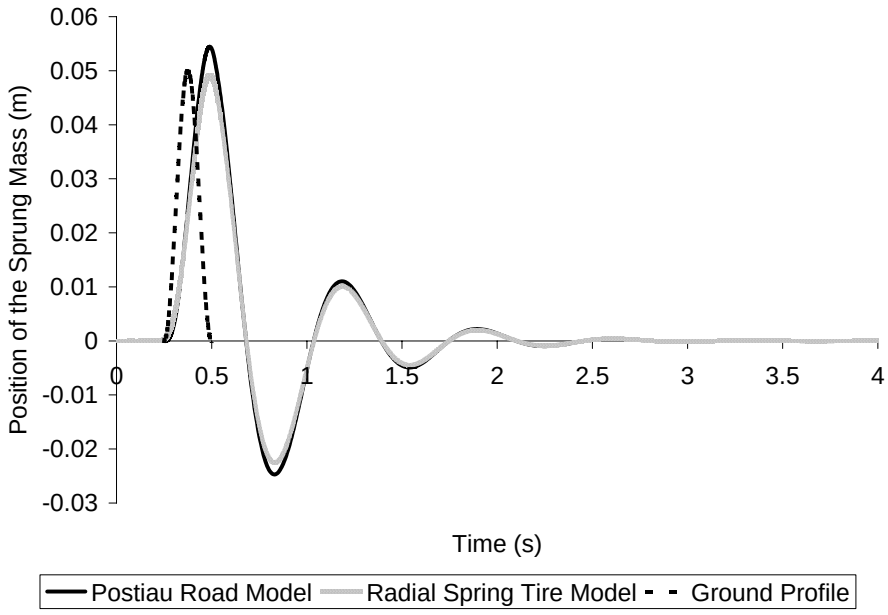


Figure 3.17: The Vertical Displacement of the Sprung Mass Versus Time

A comparison of the simulation times for a 4s simulation duration is shown in Table (3.5) as the average of 5 runs.

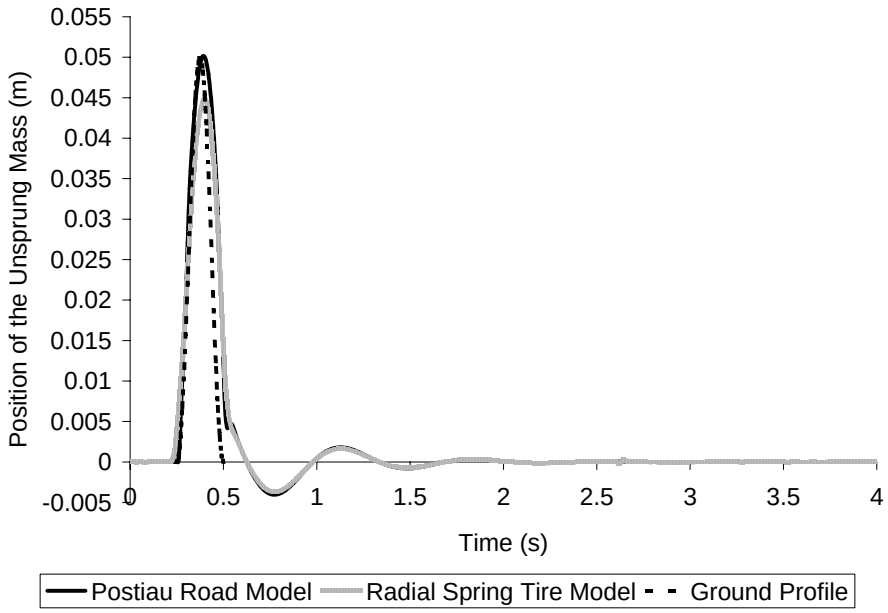


Figure 3.18: The Vertical Displacement of the Unsprung Mass Versus Time

Approach	Simulation time (s)
3D equivalent volume model	22.07
Thin disk tire model with variable radius	0.76

Table 3.5: Comparison Between the Simulation Times Obtained from both Approaches Used to Calculate the Contact Point

It can be seen that the results obtained from both approaches are in phase, but their amplitudes are different, and this is because of one of the underlying assumptions with the thin disk tire model with variable radius approach. This method assumes that there is only one point of contact with the ground. Therefore, when there is more than one point of contact with the ground, only one of them is used and not the average of them as it should be, refer to Figure (3.19). The 3D equivalent volume model takes into consideration the ground profile and abrupt changes in the ground profile. However, the thin disk tire model with variable radius approach simulates 29 times faster than the 3D equivalent volume model approach because there is only one numerical iteration (to locate the variable radius in the wheel plane) required to obtain the result. Therefore, it was decided to use the thin disk tire model with variable radius method because of its ease of implementation with a symbolic formulation approach and because of its improved simulation time. It is also important to note that the present case (vehicle going over a speed bump) would be a model used in durability studies of the vehicle and not in handling analysis of the vehicle. In handling analysis of the vehicle, abrupt changes in the road are not permitted since this will cause the frequency of the tire to be out of

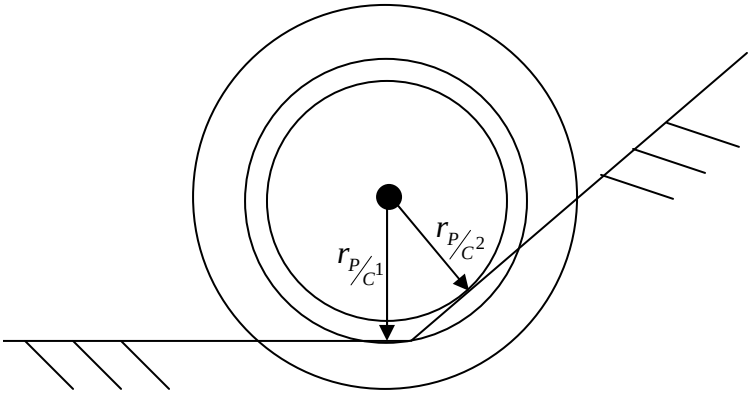


Figure 3.19: The Limiting Assumption to the Thin Disk Tire Model with Variable Radius Approach

a range that can be properly predicted by tire models [62]. Thus, the difference between the two models will be even less.

3.3.2 Thin Disk Tire Model with Variable Radius

The thin disk tire model with variable radius approach was developed by Postiau et al. [76] using a model proposed by Fisette and Samin [40]. The nomenclature that is used to describe this approach in this thesis is shown in Figure (3.20), where,

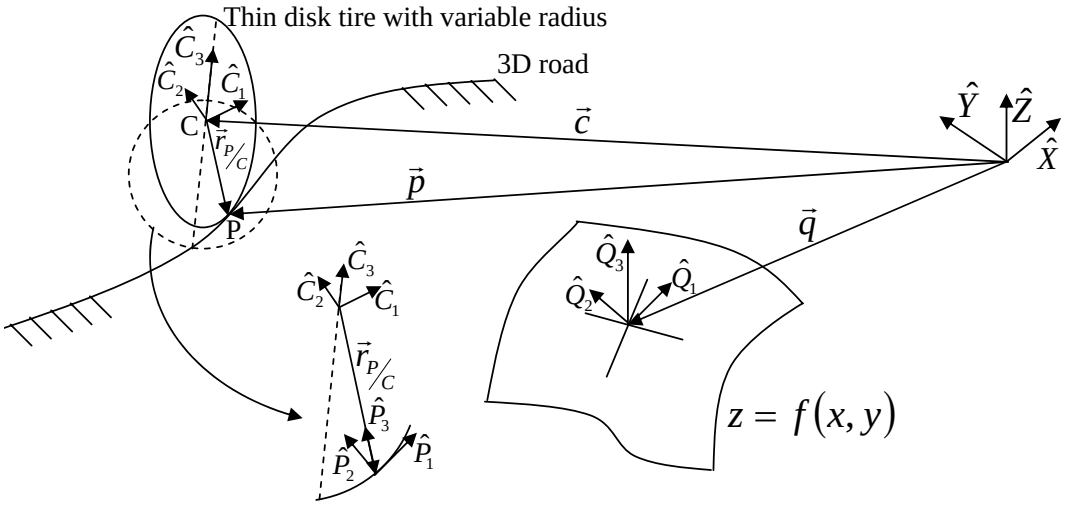


Figure 3.20: Nomenclature Used to Describe the Thin Disk Tire Model with Variable Radius Approach

the vector $\vec{c}$ locates the tire center and the vector $\vec{p}$ establishes the contact point. P is the frame at the contact point, where $\hat{P}_1$ is in the wheel plane and tangent to

the wheel, $\hat{P}_3$ is also in the wheel plane and acts along the variable radius, and $\hat{P}_2$ is chosen to make the coordinate frame right-handed. C is the frame at the tire center, where $\hat{C}_2$ is equivalent to the spin axis, and the symmetry axis of the tire, $\hat{C}_3$ is in the wheel plane and used as a reference to locate the variable radius, and $\hat{C}_1$ is chosen to make the frame right-handed. Q is the road reference frame, and is chosen to be equivalent to the inertial reference frame, i.e. $\vec{q} = 0$. The road reference frame is only used to ensure that the forces and moments are balanced — the tire forces and moments should also be reacted on the road surface. The vector $\vec{r}_{P/C}$ is the variable radius and the ground (inertial) reference frame is given by $\hat{X}$, $\hat{Y}$, and $\hat{Z}$.

The assumptions to this approach are as follows.

- The road is modeled as an elevation function ($z = f(x, y)$)
- The radius of the road is much larger than the tire radius, implying that there is only one point of contact between the tire and the ground
- The ground is assumed to be rigid and smooth
- The tire is assumed to be a planar disk with variable radius

As discussed briefly in Section (3.2.2), the contact point is obtained by varying the radius of the planar disk and its orientation in the wheel plane until it comes into contact at a single point with the road surface. This is accomplished by solving Equations (3.44) and (3.45) [76]. These equations were developed using the terminology specified above.

$$Equ_1 = \vec{p} \cdot \hat{Z} - f(\vec{p} \cdot \hat{X}, \vec{p} \cdot \hat{Y}) = 0 \tag{3.44}$$

$$Equ_2 = \hat{P}_1 \cdot \hat{u}_{ISOZ} = 0 \tag{3.45}$$

Equation (3.44) states that the height of the contact point calculated from the road profile ($z = f(x, y)$) and from the tire center (using vector $\vec{c}$ and the variable radius $\vec{r}_{P/C}$; ie $\vec{p} = \vec{c} + \vec{r}_{P/C}$) must be the same. It represents the fact that the tire must be in contact with the ground. Equation (3.45) states that the tire must be tangent to the ground at the point of contact. It forces the unit vector $\hat{P}_1$ to be equivalent to the ISO X unit vector (refer to Equation (3.5)).

The two equations need to be expressed in a form that permits them to be solved using numerical techniques. This can be accomplished using geometrical relations to update the equations so that they are a function of the magnitude of the variable radius ($w = \left|\vec{r}_{P/C}\right|$) and its orientation θ, the angle between $\hat{C}_3$ and the variable radius, as shown in Figure (3.21). Equations (3.44) and (3.45) can be updated by expressing the vector $\vec{p}$ and the unit vector $\hat{P}_1$ in terms of w and θ.

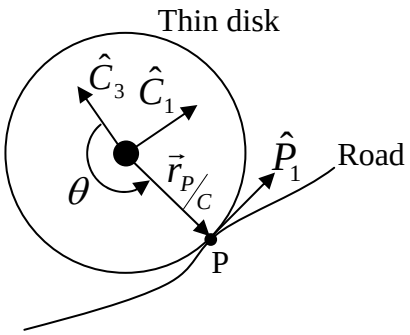


Figure 3.21: The Reference of the Orientation (θ) Used to Locate the Variable Radius

Equations (3.46)-(3.48) are used to accomplish this [24].

$$\vec{r}_{P/C} = w\cos(\theta)\,\hat{C}_3 - w\sin(\theta)\,\hat{C}_1 \tag{3.46}$$

$$\vec{p} = \vec{c} + \vec{r}_{P/C} \Rightarrow \{p\} = \{c\} + [R_C]^T \begin{Bmatrix} -w\sin(\theta) \\ 0 \\ w\cos(\theta) \end{Bmatrix} \tag{3.47}$$

$$\hat{P}_1 = -\cos(\theta)\hat{C}_1 - \sin(\theta)\hat{C}_3 \Rightarrow \{P_1\} = [R_C]^T \begin{Bmatrix} -\cos(\theta) \\ 0 \\ -\sin(\theta) \end{Bmatrix} \tag{3.48}$$

It is to be noted that the ISO Z unit vector can also be expressed in terms of the variable radius and its orientation by substituting expressions of the x and y components of the vector $\vec{p}$ from Equation (3.47) into Equation (3.4). Note, also that the equations are expressed in the ground reference frame (global frame). The rotation matrix ($[R_C]^T$) is used to transform the vectors expressed in the tire frame (frame C) to the ground reference frame so that the equations are expressed in a common frame.

3.4 Newton-Raphson Iteration Scheme

A system of two non-linear equations needs to be solved at each time step for each tire in order to determine the contact point between the tire and the road at the time of interest. These two equations can be solved using either a numerical or a symbolical approach. If a symbolic method is to be used then the equations are solved during the formulation stage allowing the result to be used during simulation to determine the contact point between the tire and the ground. On the other hand, if a numerical approach is used then the equations are developed during the formulation stage and solved during simulation using the desired method. It is important to note that the equations will be large and complex because they are both dependent on the elevation function that is used to define the road and

the height of the tire. Thus, it may take a long time to obtain an analytical solution and there may also be cases where an analytical solution may not exist. This was confirmed by solving the two equations for several different forms of the road function using Maple's *solve* package, which is a procedure embedded into Maple that uses a symbolic approach to solve systems of non-linear equations [12]. Table (3.6) shows the amount of time required to obtain a solution using Maple's *solve* package for different road profiles. The table shows that the more complex

Road function ($z = f(x, y)$)	Time to obtain expressions (s)
$z = Ax + By$	0.26
$z = Ax^2 + By^2$	54min
$z = Ax^2 + Bx + C$	11.18
$z = Ax^3 + By^3$	>5hrs
$z = Ax^3 + Bx^2 + Cx + D$	>7.8hrs
$z =$ simple piecewise function	10.2min

Table 3.6: Comparison Between the Times Required to Obtain an Algebraic Solution for the Location of the Contact Point using Maple's *solve* Package

the road function is, the longer it takes to obtain a solution and that for some road functions a solution may not even exist. Thus, using a numerical approach to solve the two equations is more desirable since a solution will always exist. However, symbolic techniques can still be utilized to optimize the numerical approach during the formulation stage since a symbolic formulation is used.

There exists several different numerical approaches that can be adopted to solve the set of non-linear equations. However, the Newton-Raphson iteration scheme is used since this approach is quadratically convergent [77]. The general form of the Newton-Raphson iteration scheme is shown in Equation (3.51).

$$\mathbf{F}(x) = \begin{pmatrix} f_1(x_1, \cdots, x_m) \\ \vdots \\ f_m(x_1, \cdots, x_m) \end{pmatrix} = 0 \tag{3.49}$$

$$\mathbf{J}(x) = \begin{bmatrix} \frac{\partial f_1(x)}{\partial x_1} & \cdots & \frac{\partial f_1(x)}{\partial x_m} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m(x)}{\partial x_1} & \cdots & \frac{\partial f_m(x)}{\partial x_m} \end{bmatrix} \tag{3.50}$$

$$\mathbf{x}_{n+1} = \mathbf{x}_n - \mathbf{J}(\mathbf{x}_n)^{-1}\mathbf{F}(\mathbf{x}_n) \tag{3.51}$$

where, $\mathbf{F}$ is a column matrix of the m equations to be solved, $\mathbf{x}$ is a column matrix of the m variables to be solved, and $\mathbf{J}$ is the jacobian matrix. Thus, for this particular case, the $\mathbf{F}$ matrix will consist of Equation (3.44) and (3.45). The Newton-Raphson method in terms of w and θ is shown in Equation (3.52).

$$\begin{Bmatrix} w_{n+1} \\ \theta_{n+1} \end{Bmatrix} = \begin{Bmatrix} w_n \\ \theta_n \end{Bmatrix} - \begin{bmatrix} \frac{\partial Equ_1}{\partial w} & \frac{\partial Equ_1}{\partial \theta} \\ \frac{\partial Equ_2}{\partial w} & \frac{\partial Equ_2}{\partial \theta} \end{bmatrix}^{-1} \begin{Bmatrix} Equ_1(w_n, \theta_n) \\ Equ_2(w_n, \theta_n) \end{Bmatrix} \tag{3.52}$$

This method is an iterative scheme and needs a necessary criteria for convergence. The error (the **F** matrix), a convergence measure, or both can be used as the criteria. The **F** matrix can be utilized by verifying that both of the equations are below a desired amount, as shown in Equations (3.53) and (3.54).

$$|\mathbf{F}_1| = |Equ_1| \leq 0.01 \tag{3.53}$$

$$|\mathbf{F}_2| = |Equ_2| \leq 0.01 \tag{3.54}$$

A measure of the convergence of the Newton-Raphson can be utilized by monitoring the result of both w and θ and comparing them to their previous results. Convergence is achieved when the difference between the two results is below a desired amount (tolerance), as is shown in Equations (3.55) and (3.56).

$$\frac{|w_{n+1} - w_n|}{w_n} \leq 0.01 \tag{3.55}$$

$$|\theta_{n+1} - \theta_n| \leq 0.01 \tag{3.56}$$

Note, a normalized amount is used to determine the convergence of w, but not for θ since the units of θ will always be in radians. The latter approach will be adopted, with the tolerance shown in Equations (3.55) and (3.56) to determine the convergence criteria of the Newton-Raphson iteration since it is not required to use both methods.

The equations usually converge within 2 to 4 iterations using either criteria. It is also important to note that an initial guess needs to be specified to the Newton-Raphson method, and if the initial guess is not close to the solution the system may diverge and not converge. Thus, it is important to have a proper initial guess when using this scheme. Nonetheless, this method is used to obtain the contact point between the tire and the road surface for each tire at each time step.

3.5 Road Model Types

There are many different road profiles that can not be modeled by a single elevation function. It is also very difficult to match experimental road data to a single function that represents the road properly. Thus, in cases where the road can not be modeled as a single elevation function a set of tabular data is used to represent the road. A spline is utilized to create the elevation function between two sets of data points, where each data point consists of a location on the centerline of the road with respect to the cartesian (global) reference frame. It is common practice to represent the road as a set of x, y, and z data points that represent the centerline of the road [78, 16], and this method is used by MSC.ADAMS [3] and CarSim [2]. The distance traveled along the path (centerline of the road), s, is used as the independent variable in the spline function, allowing the spline to be a one-dimensional spline [16, 45, 27]. There are also many cases where the road plane varies only in the x direction and just banks with respect to the y direction.

A banked road is defined by the $z = 0$ road plane that is rotated with respect to the local x axis of the road. In this case a piecewise spline is used to represent the data and the x position is used as the independent variable in the spline functions. Figure (3.22) shows the flow diagram that is to be followed when choosing the desired method to define the road. Note that α is the bank angle. It can also be

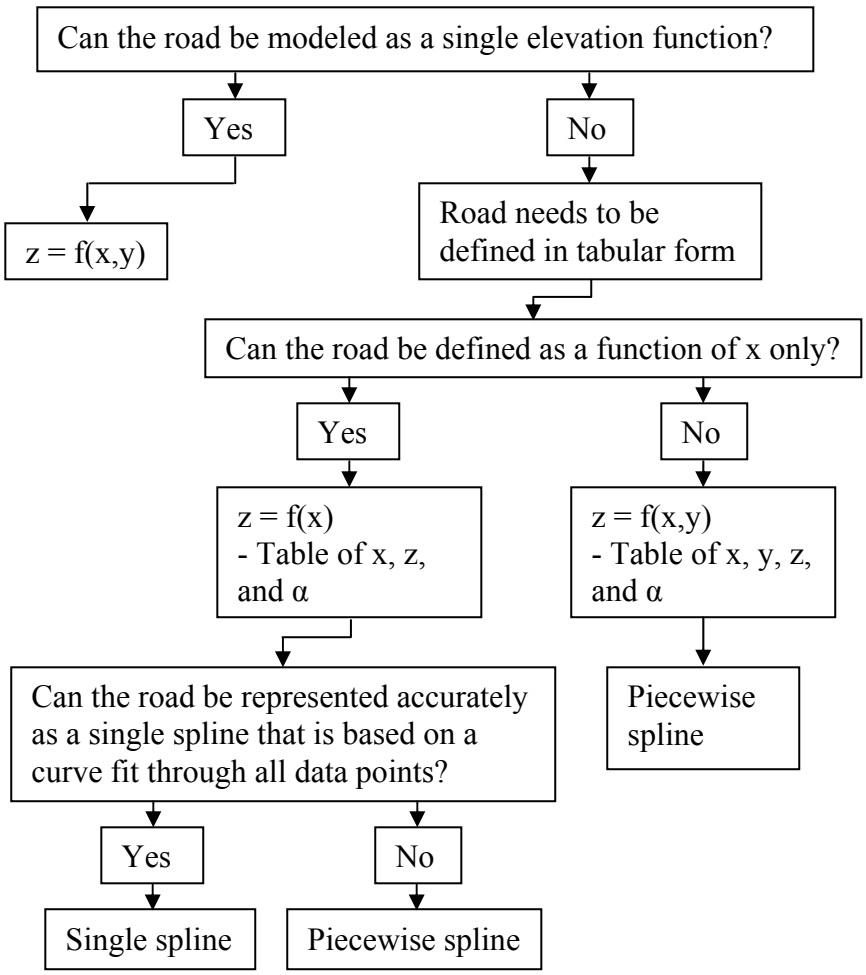


Figure 3.22: The Flow Diagram for the Road Model Type

noted that a single spline can be used to represent the road — a single spline is a single curve fit to all of the data [46]. This option is only available when the road varies with respect to x only, since there are only a few cases (like a hill for example) that can be represented properly in this manner.

Once the tabular data is known, the distance traveled along the path is calculated using Equation (3.57).

$$s_{n+1} = s_n + \sqrt{(x_{n+1} - x_n)^2 + (y_{n+1} - y_n)^2 + (z_{n+1} - z_n)^2} \tag{3.57}$$

The spline coefficients are determined based on the desired spline order. Throughout this thesis a cubic spline will be used to represent the road between two data

points. The cubic spline is chosen because it permits the second derivative to be continuous throughout the road profile [16, 77]. However, a linear spline should be used if the cubic spline function causes the road profile to vary too much from the straight line between two data points. If this is the case then the distanced traveled along the path (as per Equation (3.57)) will inadequately indicate the current section of the road. Also, when a cubic spline function is used the second derivative is required as input in order to calculate the spline coefficients. It can be specified in the table representing the road, or can be approximated by solving a system of equations given the value of its second derivative at two data points as input [77]. Throughout this thesis the second derivative is assumed to be known ahead of time and specified in the table as input. The cubic spline coefficients are given in Equations (3.58)-(3.62), and shown in Figure (3.23).

$$z = f(s) = ASs^3 + BSs^2 + CSs + DS \tag{3.58}$$

$$AS = \frac{\left(\frac{\partial^2 z}{\partial s^2}_{n+1} - \frac{\partial^2 z}{\partial s^2}_{n}\right)}{\left(6\left(s_{n+1} - s_n\right)\right)} \tag{3.59}$$

$$BS = \frac{\left(\left(s_{n+1}\right)\left(\frac{\partial^2 z}{\partial s^2}_{n}\right) - \left(s_n\right)\left(\frac{\partial^2 z}{\partial s^2}_{n+1}\right)\right)}{\left(2\left(s_{n+1} - s_n\right)\right)} \tag{3.60}$$

$$\Delta z = \left(z_{n+1} - z_n\right)$$

$$\Delta s = \left(s_{n+1} - s_n\right)$$

$$CS = \frac{(6\Delta z) + \left(\left(\frac{\partial^2 z}{\partial s^2}_{n}\right)\left((\Delta s)^2 - \left(3\left(s^2_{n+1}\right)\right)\right)\right) + \left(\left(\frac{\partial^2 z}{\partial s^2}_{n+1}\right)\left(\left(3\left(s^2_{n}\right) - \left((\Delta s)^2\right)\right)\right)\right)}{(6\Delta s)} \tag{3.61}$$

$$DS = \frac{6\left(\left(s_{n+1}\right)\left(z_n\right) - \left(s_n\left(\left(z_{n+1}\right)\right)\right) + \frac{\partial^2 z}{\partial s^2}_{n}\left(s^3_{n+1} - \left((\Delta s)^2\right)s_{n+1}\right) + \frac{\partial^2 z}{\partial s^2}_{n+1}\left(\left(\left((\Delta s)^2\right)s_n\right) - s^3_n\right)\right.}{(6\Delta s)} \tag{3.62}$$

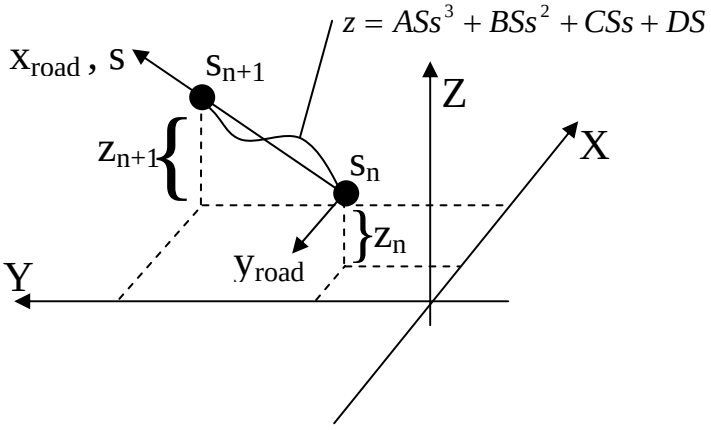


Figure 3.23: The Nomenclature Used to Describe the Cubic Spline

The road can also be banked with respect to its x axis, x_{road}, to allow its height to vary along the y axis of the road, y_{road} [94], and is shown in Equation (3.63) and

in Figure (3.24).

$$z = (\tan(\alpha))\, y_r \tag{3.63}$$

where, y_r is the distance of interest along y_{axis}, as shown in Figure (3.24). The

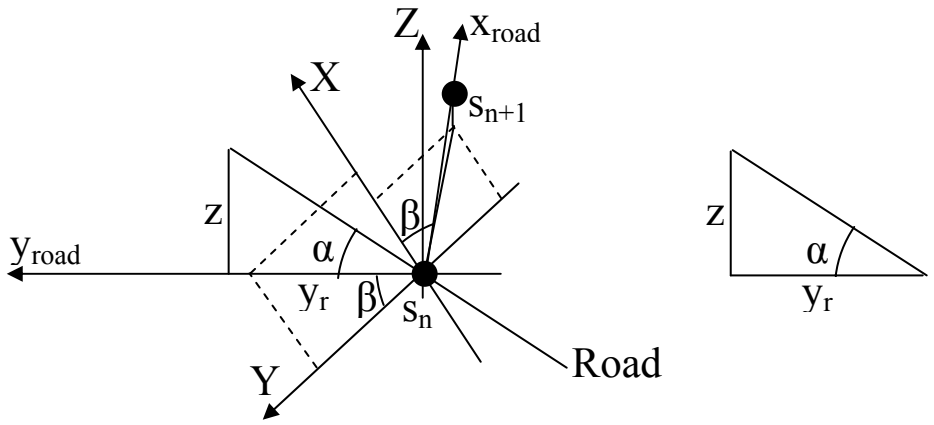


Figure 3.24: The Nomenclature Used to Describe the Bank Angle

bank angle for each section of the road (between two data points) is also specified as input in the table. It is to be noted that the y_{road} axis lies in the XY plane, and that a simple rotation by β about the global Z axis can be used to rotate y_r to the global frame. This permits the bank angle to be specified with respect to the global frame, as shown in Equation (3.64).

$$z = (\tan(\alpha))\,(-\sin(\beta)\,(x - x_n) + \cos(\beta)\,(y - y_n)) \tag{3.64}$$

$$\beta = \arctan\left(\frac{(y_{n+1} - y_n)}{(x_{n+1} - x_n)}\right) \tag{3.65}$$

where x is the current x position of the tire, y is the current y position of the tire, and β can be calculated from the data in the table representing the road. It can be noted that the data specified in the table is with respect to the global reference frame. Note in most circumstances the axis of the road frame will not be aligned with the global axis and thus a translation between frames is also required, as shown in Equation (3.64).

The road functions are combined by superposition to give the final form of the elevation function that represents the road between two data points, as shown in Equation (3.66).

$$\begin{aligned} z &= ASs^3 + BSs^2 + CSs + DS \\ &\quad + (\tan(\alpha))\,(-\sin(\beta)\,(x - x_n) + \cos(\beta)\,(y - y_n)) \qquad (3.66) \\ &= z_1 + z_2 \qquad (3.67) \end{aligned}$$

The current distance traveled by the vehicle is used to determine the current elevation of the road at each of the axles of the vehicle in order to determine the

current cubic spline coefficients for each tire. Note, the current distance traveled by the axle (s_{axle}) needs to be used in Equation (3.66) to determine z_1, and not the current distance traveled by the tire as is used to calculate z_2. Refer to Equation (3.68) and Figure (3.25) for an example of how this is done for the front axle of a passenger vehicle.

$$s_{axle} = s + s_{front\_axle} \tag{3.68}$$

Note, the top view is used in Figure (3.25) for clarification purposes. The axle

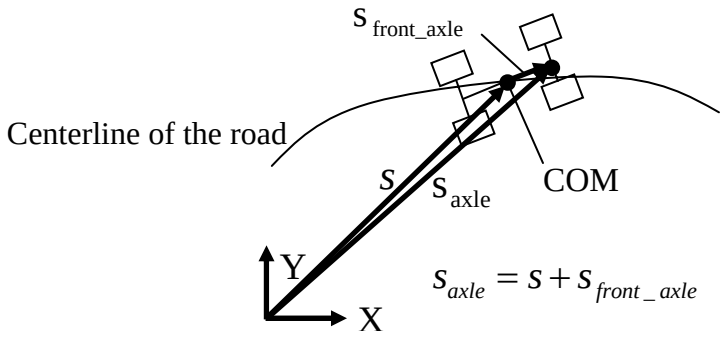


Figure 3.25: The Distance Traveled Along the Path of the Road by a Front Tire

position may not be on the centerline of the road as is shown in Figure (3.25), but will be close to it since drastic changes in the road are not permitted nor are they realistic. If there is a bank angle specified than the current position of the tire along with the current bank angle are used to adjust the elevation of the road appropriately.

The thin disk tire model with variable radius outlined in Section (3.3) can still be used to determine the contact point between the tire and the ground. However, when determining the surface normal it is important that the chain rule from calculus is used. As an example, this is shown in Equation (3.70) for the partial derivative of z with respect to x.

$$\frac{\partial z}{\partial x} = \frac{\partial z_1}{\partial x} + \frac{\partial z_2}{\partial x} = \left(\frac{\partial z_1}{\partial s}\right)\left(\frac{\partial s}{\partial x}\right) + \frac{\partial z_2}{\partial x} \tag{3.69}$$

$$\frac{\partial z}{\partial x} = 3ASs^2+2BSs+CS\frac{x_{n+1}-x_n}{\sqrt{(x_{n+1}-x_n)^2+(y_{n+1}-y_n)^2}}-\tan(\alpha)\sin(\beta) \tag{3.70}$$

Note, that the second partial derivative term ($\partial z_1/\partial x$) in the equation is to account for the direction of the surface normal to ensure that it is properly defined with respect to the global reference frame. It is also equivalent to rotating the first term by β with respect to the global Z axis, as shown in Figure (3.24) and in Equation (3.71).

$$\cos(\beta) = \frac{x_{n+1} - x_n}{\sqrt{(x_{n+1} - x_n)^2 + (y_{n+1} - y_n)^2}} \tag{3.71}$$

A similar derivation to that shown previously can be done for the case when the road only varies in the x direction. In fact, this is a special case of the equations presented, where s becomes a function of x and z only since the centerline of the road always lies along the $y = 0$ position.

The thin disk tire model can still be used to calculate the contact point since the road is still defined as a function, but in sections instead. It can also be noted that it is also common to use a driver model when the centerline of the road is specified to force the vehicle to follow the predefined path [23, 61]. Therefore, tabular methods are used to define the road when it can not be represented as a single elevation function.

Chapter 4

Symbolic Computer Implementation

4.1 Overview

Two equations have been obtained to solve for the contact point between the tire and the ground. These equations are solved using the Newton-Raphson iteration scheme (a numerical scheme) for each tire at each time step. However, symbolic operations can be used to improve and prepare the equations for simulation because a symbolic formulation approach is utilized. Modifications to them can be performed using symbolics to decrease the time required to solve the equations. The method is implemented in Maple, which is a general-purpose computer algebra package. Maple is created by Maplesoft, a leading provider for high-performance software tools for engineering, science and mathematics [12].

Based on the modified equations, a simulation code structure is created to permit the forces and moments that are applied to the vehicle model (F_C and M_C from Figure (3.1)) to be evaluated at each time step for each tire. This code structure is based on the steps necessary to include a tire on a 3-D road in multibody dynamic models of the vehicle, as outlined in Section (3.2). During the formulation stage, the simulation code structure is used to create a number of procedure calls that allow the forces to be evaluated through intermediate stages, such as determining the contact point between the tire and the ground, and calculating the tire forces and moments using a tire model.

Methods to improve the two equations using symbolics is discussed first, followed by the introduction of the hardware float environment and simulation code optimization. The development of the simulation code structure is defined and discussed. The chapter ends with a brief discussion of how the structure is integrated in the DynaFlexPro software package, which automatically generates the governing equations given the description of the system as input [8].

4.2 Developments of the Newton-Raphson Iteration Routine

A full numerical approach is one where the equations are reconstructed within the Newton-Raphson method at each iteration before they are solved. However, a full numerical approach is not required since the approach is implemented in a symbolic environment. Thus, symbolic methods can be used to first construct the equations and then use them in the numerical iteration to solve for the contact point. This eliminates the equations from having to be re-formulated inside each iteration. Further improvements/developments can be made to the equations to better prepare them for simulation using symbolic techniques.

4.2.1 Custom Approach Versus Maple's *fsolve*

The two equations can also be solved using Maple's *fsolve* package, which solves systems of non-linear equations based on a combination of root location and numerical techniques like the Newton-Raphson method [29]. However, *fsolve* does many different manipulations to the equations before they are solved and uses more than one method to determine the result [12] — many of these additional modifications are not required when there are only 2 equations to be solved and the result is a real number and not an imaginary one. Thus, it was decided to create a custom approach based on the Newton-Raphson method and to compare the time required to obtain the result to that of *fsolve* for different road profiles. The initial position and orientation of the tire frame (frame C) of the front left tire of the Chevrolet Equinox is used as input to the two equations, and is shown in Equations (4.1) and (4.2).

$$\{c\} = \begin{Bmatrix} 1.353 \\ 0.76 \\ 0.35 \end{Bmatrix} \tag{4.1}$$

$$[R_C] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.2}$$

The only other necessary information to solve the equations is the road profile that is required to generate them and an initial guess. It was decided to use an initial guess of $w = -R_u = -0.355$m and $\theta = 0$rad since the tire is initially aligned with the global frame. Table (4.1) shows a comparison of the times required to obtain a solution from the two different approaches for three different road profiles. Note, the times recorded in the table are the average of 5 runs. Note, for the second case, the position of the front left tire was modified appropriately to account for the vehicle being inclined with the road profile. Also, for both cases the equations were developed first and then used in the iteration scheme for the custom approach and as input to *fsolve*.

Road function	Maple fsolve (s)	Custom approach (s)
$z = 0$ (flat surface)	0.39	0.28
$z = x$ (45deg inclination)	0.60	0.15
$z = \begin{cases} 0 & : \quad x \leq 5 \\ (x-5) & : \quad x > 5 \end{cases}$ (Combination of both)	1.27	1.26

Table 4.1: Comparison Between the Times Required to Obtain a Solution for the Contact Point Using the Custom Approach and Maple's fsolve Package

The time required to obtain the solution is faster when using the custom approach than using the embedded approach in Maple. It can be noted that the result obtained from both methods was the same for the three cases considered. Note, for both cases the software float environment was used by Maple to determine the solution. However, significant gains can be achieved in the simulation time if the hardware float environment is used over the software float environment [55]. This is only achievable with the custom approach since *fsolve* will always operate in the software float environment. Also, the equations can not be exported when *fsolve* is used since it is an embedded procedure within Maple. Refer to Section (4.3) for details on the software and hardware float environments.

Therefore, the custom approach is used as the method to solve for w and θ at each time step. Additional improvements can be made to the custom approach by optimizing the equations before they are solved. Doing this will lead to further improvements in the time required to obtain a result.

4.2.2 Prepare the Equations for Simulation and Pre-Compute the Inverse of the Jacobian

The number of mathematical operations performed in each iteration to calculate w and θ can be decreased by developing the equations before the Newton-Raphson iteration scheme. This is similar to the in-line integration approach used in Dymola [72]. Thus, the results are used during the iterations by making substitutions into the equations, specifically for Δw and $\Delta\theta$, which are shown in Equations (4.3) and (4.4).

$$w_{n+1} = w_n + \left\{ \left[\frac{\partial Equ_1}{\partial \theta} \frac{1}{det(J)} Equ_2 \right] - \left[\frac{\partial Equ_2}{\partial \theta} \frac{1}{det(J)} Equ_1 \right] \right\} \tag{4.3}$$

$$w_{n+1} = w_n + \Delta w$$

$$\theta_{n+1} = \theta_n + \left\{ \left[\frac{\partial Equ_2}{\partial w} \frac{1}{det(J)} Equ_1 \right] - \left[\frac{\partial Equ_1}{\partial w} \frac{1}{det(J)} Equ_2 \right] \right\} \tag{4.4}$$

$$\theta_{n+1} = \theta_n + \Delta\theta$$

Therefore, within the Newton-Raphson scheme, the equations are used directly without requiring any development. It can be noted that the two equations can

be hard-coded if the road function is known ahead of time, in which case the equations do not have to be reformulated at each time step before the Newton-Raphson scheme is used to determine the contact point; resulting in a decrease in the simulation time.

It is the inverse of the Jacobian matrix that requires the most computations to perform when formulating the two equations. It can be obtained using an LU decomposition approach [41] or using the known solution — a solution to the inverse of the Jacobian is known since it is a 2 × 2 matrix [99], and is shown in Equation (4.5).

$$J(w,\theta)^{-1} = \frac{1}{det(J)} \begin{bmatrix} \frac{\partial Equ_2}{\partial \theta} & -\frac{\partial Equ_1}{\partial \theta} \\ -\frac{\partial Equ_2}{\partial w} & \frac{\partial Equ_1}{\partial w} \end{bmatrix} \tag{4.5}$$

$$\text{where:} \qquad det(J) = \frac{\partial Equ_1}{\partial w}\frac{\partial Equ_2}{\partial \theta} - \frac{\partial Equ_1}{\partial \theta}\frac{\partial Equ_2}{\partial w} \tag{4.6}$$

The equations can be developed using the LU decomposition or the known solution instead of calling Maple's external function to calculate the inverse. It is to be noted that the simulation/formulation time is also decreased by eliminating the number of function calls [97]. Thus, the number of operations required to obtain the final result for Δw and $\Delta \theta$ using both of the two approaches is compared, and the results are summarized in Table (4.2).

Operation	LU decomposition	Known solution to the matrix inverse
multiplication	5	6
division	5	4
addition	1	2
subtraction	2	1
Total	13	13

Table 4.2: Comparison Between the Number of Operations Required to Formulate the Two Equations Using the LU Decomposition Approach and the Known Solution to the Inverse of a 2 × 2 Matrix

It can be seen that the number of operations required to obtain the equations are the same using either of the approaches. Therefore, the known solution to the matrix inverse will be adopted to formulate Δw and $\Delta \theta$ because of its ease of implementation. It can be noted that the Δw and $\Delta \theta$ presented in Equations (4.3) and (4.4) were developed using the known solution to the inverse of the Jacobian (Equation (4.5)).

4.2.3 Convergence of the Newton-Raphson Iteration Scheme

The main disadvantage of using the Newton-Raphson method is that the inverse of the Jacobian is required to solve the two equations and when $det(J)$ is close to

zero, the Jacobian becomes ill-conditioned. When it is ill-conditioned the system of equations may take a long time to converge or may even diverge [53]. Depending on the initial guess of w and θ, the system of equations may diverge. Thus, the conditioning of the determinant of the Jacobian is dependent on the initial guess. It is to be noted that the most common initial guess used for the Newton-Raphson iteration scheme is to use the previous result for the current iteration [46]. The determinant of the Jacobian is equal to zero when the value of w and θ in Equation (3.45) is such that the $\hat{P}_1$ unit vector is parallel to the surface normal $\hat{u}_{ISOZ}$, as shown in Figure (4.1). In this circumstance, the dot product of the two vectors

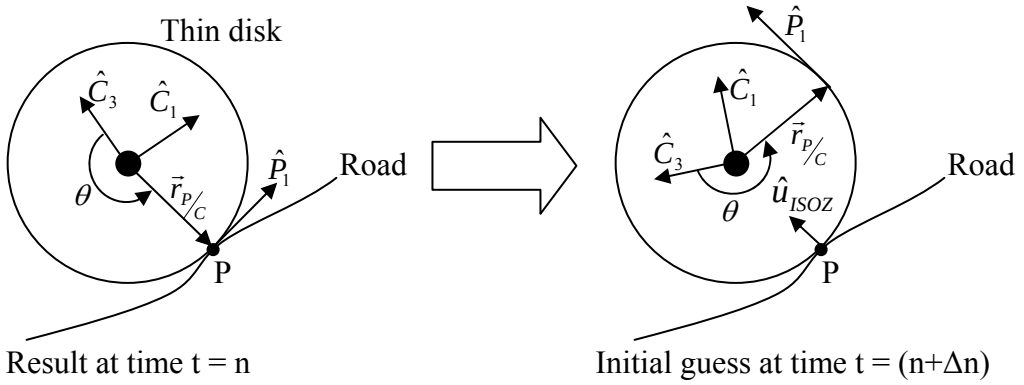


Figure 4.1: Case when the Determinant of the Jacobian can be Equal to Zero

can not be equal to zero [99]. If the tire rotates by a certain amount in one time step and the previous result is used as the initial guess to the Newtom-Raphson algorithm, it may cause the inverse of the Jacobian to be ill-conditioned since it may cause the two vectors to be close to parallel, as shown in Figure (4.1). It is to be noted that it is the value of θ that causes the Jacobian to be ill-conditioned and not the value of w. Therefore, there needs to be a means to ensure that the Jacobian is not ill-conditioned so that the scheme always yields a valid result.

An ill-conditioned Jacobian can be avoided by doing one of the following.

- Monitor the determinant of the Jacobian and change the initial guess when appropriate
- Use a non-spinning tire frame

A non-spinning tire frame is a frame on the geometrical wheel, and thus moves with the tire, but does not rotate (spin) with it. With this approach the unit vector $\hat{C}_3$ (the reference for θ) remains in the wheel plane, but does not spin with the tire. This method will prevent the Jacobian from being ill-conditioned in most cases since the tire will pitch with the vehicle and the pitch of the vehicle is directly related to the road profile [43]. However, the initial guess is still important to the Newton-Raphson method and may still cause the Jacobian to be ill-conditioned

even if a non-spinning frame is used. Thus, both methods are utilized to prevent the Jacobian from being ill-conditioned.

The conditioning of the Jacobian was determined by performing a series of simulations by varying the initial guess of θ and monitoring the value of the determinant of the Jacobian. The criteria was chosen by monitoring the time and the number of iterations required to obtain a result. It was found that if $det(J) \geq 0.8$, the system of equations will converge in 1 to 3 iterations based on the error defined in Equations (3.55) and (3.56). Below this limit the equations will take more than 3 iterations to converge if they converge at all. Therefore, if $det(J) < 0.8$ then the initial guess of θ is changed by 90° to ensure the conditioning of the Jacobian, as shown in Figure (4.2).

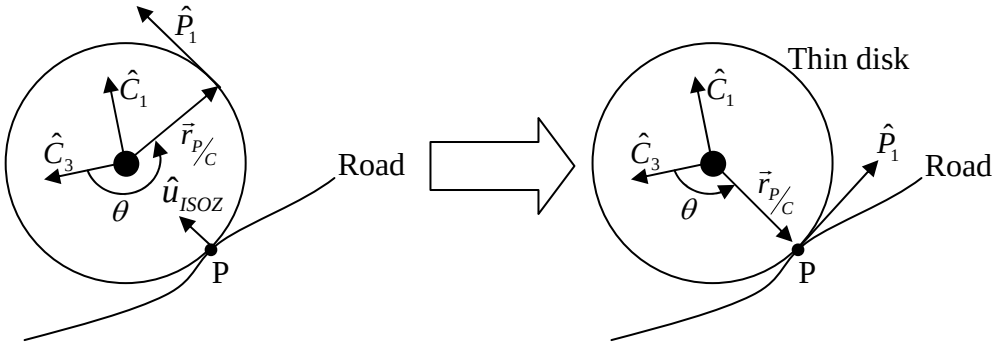


Figure 4.2: Initial Guess of θ Changed to Ensure that the Jacobian is Not Ill-Conditioned

The convergence of the Newton-Raphson method is dependent on the initial guess, and when it is close enough to the solution, the system of equations will converge in 2 iterations based on the convergence measure defined in Equations (3.55) and (3.56). It is not beneficial to use the previous result as an initial guess to the Newton-Raphson iteration since when a variable time step integrator is used, the integrator will repeat previous time steps in order to achieve the desired tolerance. When this occurs the result from the previous time step may not be close enough to the solution. Thus, it is best to always use the same initial guess and to monitor the initial guess of θ so that the determinant of the Jacobian remains greater than 0.8. In the majority of the cases, the road profile will be similar to that of a flat road and the orientation of the non-spinning frame will be such that $\hat{C}_3$ is close to pointing in the same direction as Z from the reference frame (assuming the z position of the chassis is aligned with the global Z axis). Note, the geometrical wheel will pitch with the vehicle, thus $\hat{C}_3$ will remain close to the surface normal of the road profile. Therefore, a good initial guess to use is $\theta = 0$ and $w = -R_u$, as shown in Figure (4.3). Thus, by using a non-spinning frame and monitoring the determinant of the Jacobian, and changing the initial guess of θ when appropriate, the Newton-Raphson method should yield a valid result.

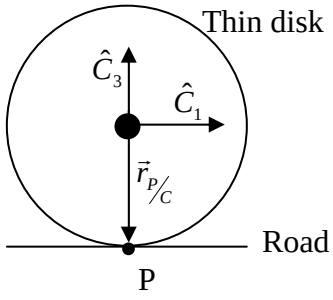


Figure 4.3: Example of the Fixed Initial Guess

4.3 Hardware Float Environment

There are two types of float environments for evaluating functions and performing operations in Maple and these are the software and the hardware float environments [12]. A hardware float environment uses hardware floats at a desired precision to evaluate the function/expression/procedure by converting the floating point numbers to floating point arguments and passing them to functions to perform the operations [55]. On the other hand, a software float environment will convert the floating point operations into a number of function calls, and use predefined libraries to perform the operations — the operations are all performed in software [55]. Symbolic operations are high-precision ones and thus require a lot of time to perform them. Maple, unfortunately can not handle both environments simultaneously due to the different floating point argument passing conventions between the two approaches [28]. Thus, if a symbolic computation is required, then the software float environment is used for all computations and this causes the simulation to be much slower than in the hardware float environment.

Maple will automatically use the high-precision computational environment when there is only one symbolic operation required [28]. Therefore, if the two equations necessary to calculate the contact point are formulated within the road model at each time step for each tire, Maple will use the software float environment to perform all of the operations — even though the majority of the operations could be performed in the hardware float environment. The road model is a procedure built around the Newton-Raphson iteration scheme to determine the contact point between the tire and the ground, and is used to compare the two different float environments for three different road profiles. Two Maple sub-routines were created for one tire at one time step, where the first sub-routine calculates the two equations and then performs the numerical iterations to calculate the contact point using the calculated equations. The two equations are hard-coded into the second sub-routine allowing the numerical iterations to be performed directly based on the current position and orientation of the tire, which is taken as the front left tire of the Chevrolet Equinox (refer to Equations (4.1) and (4.2)). The results are summarized in Table (4.3) with the simulation times being the average of 5 runs.

Road function	Software float environment (s)	Hardware float environment (s)
$z = 0$ (flat surface)	0.031	0.015
$z = \sin(x)$ (hill)	0.032	0.016
$z = x^2 + y^2$	0.047	0.016

Table 4.3: Comparison Between the Times Required to Obtain a Solution for the Contact Point Using the Software Float and Hardware Float Environments

As can be seen from the table there is about a 200 to 300% increase in the time required to obtain a result when the software float environment is used over the hardware float environment. It can also be noted that a more significant decrease in the simulation time will be noticed when the complete vehicle is simulated for more than one time step. Thus, all symbolic operations should be eliminated during simulation so that Maple can function in the hardware float environment resulting in a significant decrease in the simulation time.

The road model function needs to be pre-defined (hard-coded) in order to be able to force Maple to operate in the hardware float environment. This requires that a road model be created for each road profile that one wishes to analyze. However, Maple functions can be used during the formulation stage to automatically create the road model based on the desired road profile to eliminate all symbolic operations during simulation. This is accomplished using the symbolic operations during the formulation stage to construct both Δw and $\Delta\theta$ and then build a procedure around them that is based on the Newton-Raphson iteration to calculate the contact point. It is to be noted that the road model also determines the ISO unit vectors, the vector $\vec{r}_{P/C}$, and the loaded radius and sends them back to the vehicle model. Thus, the procedure that is created also includes the equations necessary to determine the desired outputs from the road model. The steps necessary to accomplish this during the formulation stage are outlined below.

1. Calculate Δw, $\Delta\theta$, and $detJ$ ($det(J)$) based on the road function $z = f(x, y)$, and combine the results into a single column vector by assigning each equation to a component of NR_OUT; refer to Figure (4.4)

2. Generate the ISO unit vectors, the vector $\vec{r}_{P/C}$, and the loaded radius and combine the results into a single column vector by assigning each equation to a component of NRIteration; refer to Figure (4.5)

3. Create the first portion of the middle of the procedure as a string by specifying that all three of the components of the first column vector (all three of the NR_OUTs) be assigned to their appropriate equations; refer to smidle1 in Figure (4.6)

4. Create the next portion of the middle of the procedure as a string by specifying the criteria for the determinant to ensure that the Jacobian is not

ill-conditioned; refer to smidle2 in Figure (4.6)

5. Develop the third portion as a string by creating the Newton-Raphson method utilizing the result for Δw, $\Delta\theta$, and by specifying the criteria for convergence; refer to smidle3 in Figure (4.6)

6. Formulate the final portion of the middle part as a string by specifying that all of the components in the second column vector (all of the NRIterations) be assigned to their appropriate equation; refer to smidle4 in Figure (4.6)

7. Create the top portion of the procedure as a string by assigning the position and orientation of the tire center to their appropriate variables and by initializing a guess for w and θ; refer to s_top in Figure (4.6)

8. Create the end portion of the procedure as a string by ending the procedure; refer to sbottom in Figure (4.6)

9. Parse all of the steps together to create the final road model procedure; refer to Figure (4.6)

Note that the procedures can be hard-coded for the cases when the road is represented in tabular form since the functions that represent the road are known ahead of time. Thus, the simulation time can be greatly improved by allowing Maple to run in the hardware float environment during simulation.

4.4 Simulation Code Optimization

The optimization and expression manipulation routines in Maple can be used to optimize the code for simulation. Maple has several expression manipulation routines embedded into it that can be used to simplify the expressions in order to decrease the computational cost to evaluate them [71]. Two of the most popular routines are *simplify()* and *combine()* [12], where they both use pre-defined rules for simplifying radicals, exponential functions, logarithmic functions, trigonometric functions, and various special functions [12]. The most common simplification necessary to be performed on the equations seen in multibody dynamics is of trigonometric type [70].

The *combine()* command will apply appropriate transformations to combine terms in sums, products and powers into one term [12]. Adversely, the *simplify()* routine will look in the expressions for function calls (a function call is when Maple needs to call a routine to evaluate a portion of the expression, such as $\sin(x)$), square roots, radicals and powers and tries to apply an appropriate procedure to simplify the expression [12]. An example of each of the routines is shown in Table (4.4), and it can be seen that *combine()* works best for the first example, whereas *simplify()* works best for the second example. It can also be seen in the second example that these routines can increase the computational cost associated with evaluating the

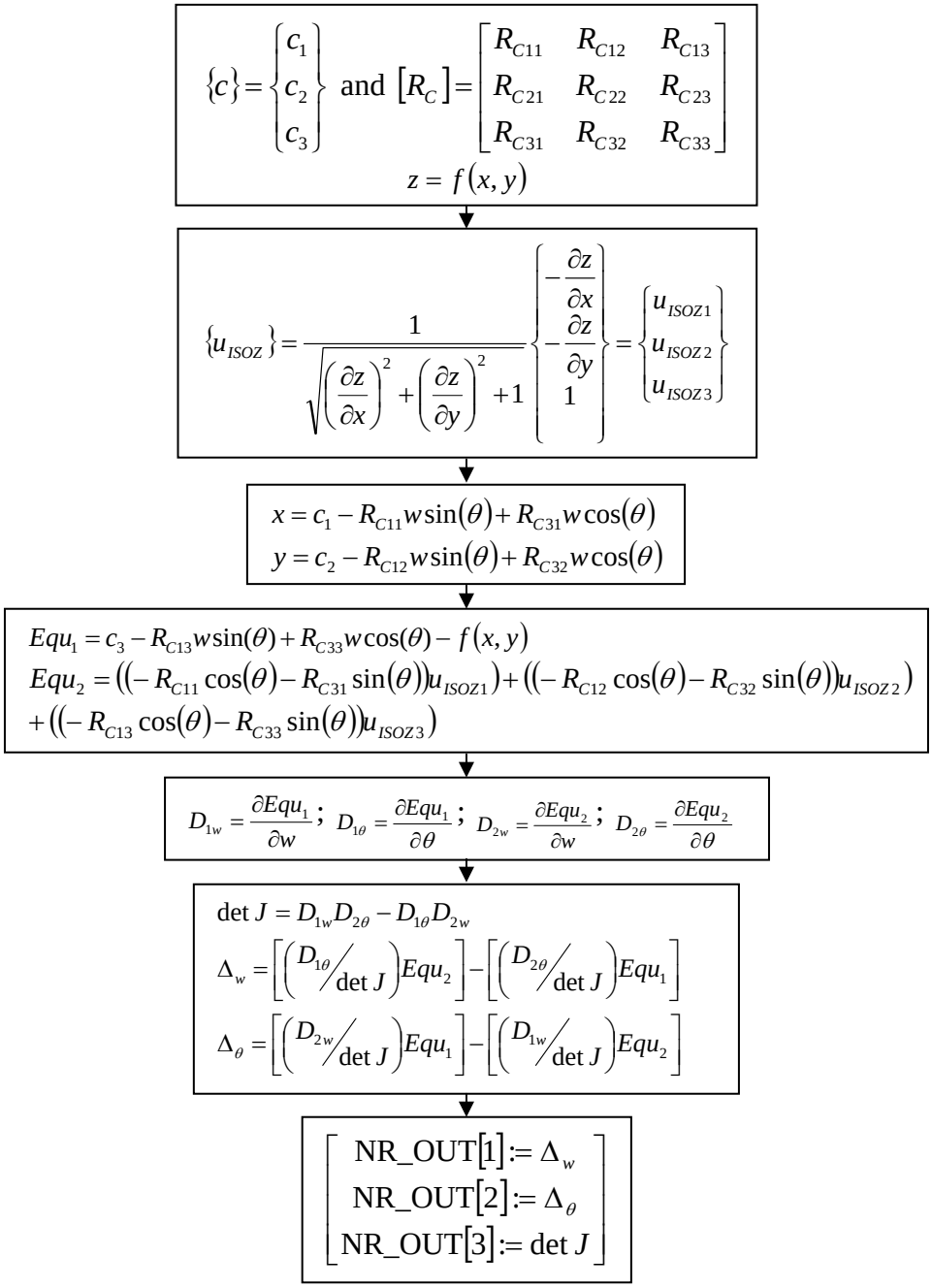


Figure 4.4: The Flow Diagram to Build Δw, $\Delta\theta$, and $detJ$ which are Necessary in the Construction of the Road Model Procedure

expression. Thus, the utility of each of the routines depends on the expression to be evaluated.

Maple has an optimization package, CodeGeneration, that will minimize the computational cost associated with evaluating a procedure [102]. The CodeGeneration package is different from Maple's expression manipulation routines since it

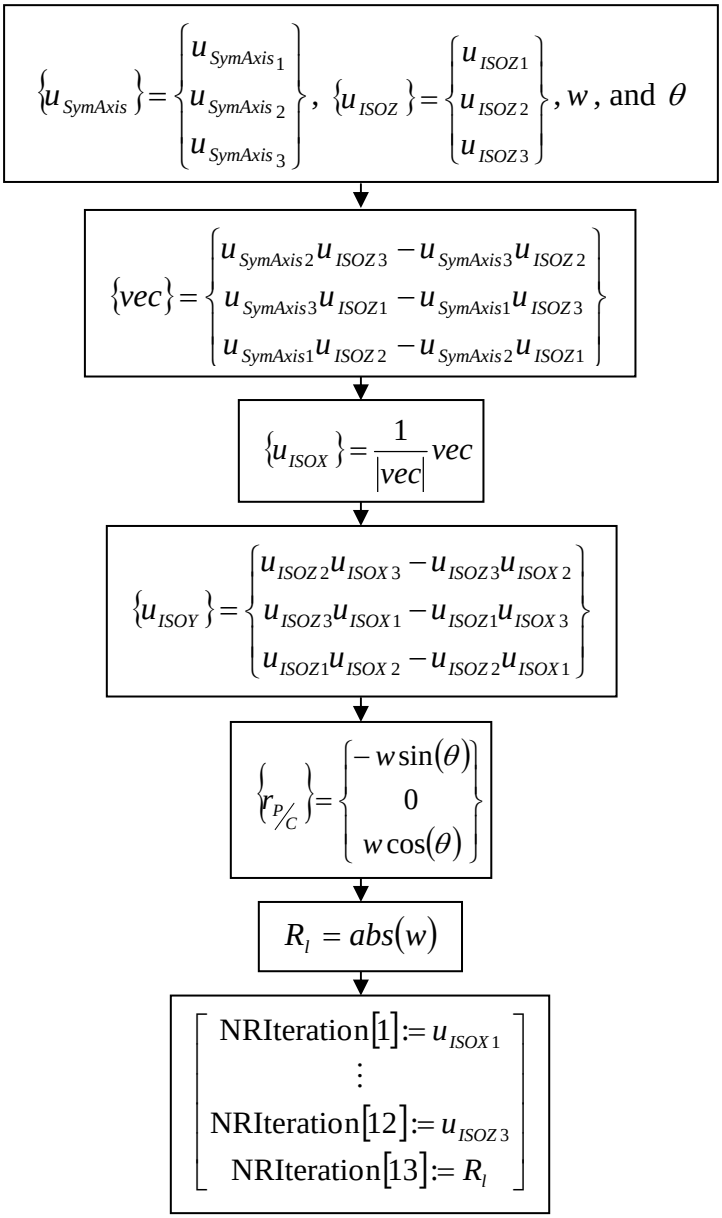


Figure 4.5: The Flow Diagram to Build the Outputs from the Automatically Generated Road Model Procedure

concentrates on optimizing a procedure versus an expression. It can still be applied to optimize an expression, but is designed to optimize a procedure. It will identify repeated terms within the procedure and will store them as temporary variables allowing them to only have to be evaluated once and used several times [102]. During the optimization, variables that are evaluated somewhere in the procedure that do not contribute to the evaluation of the output (or outputs) are dropped.

An example of the CodeGeneration package is shown in Table (4.5) to better illustrate how the package functions. The procedure defined takes the values of

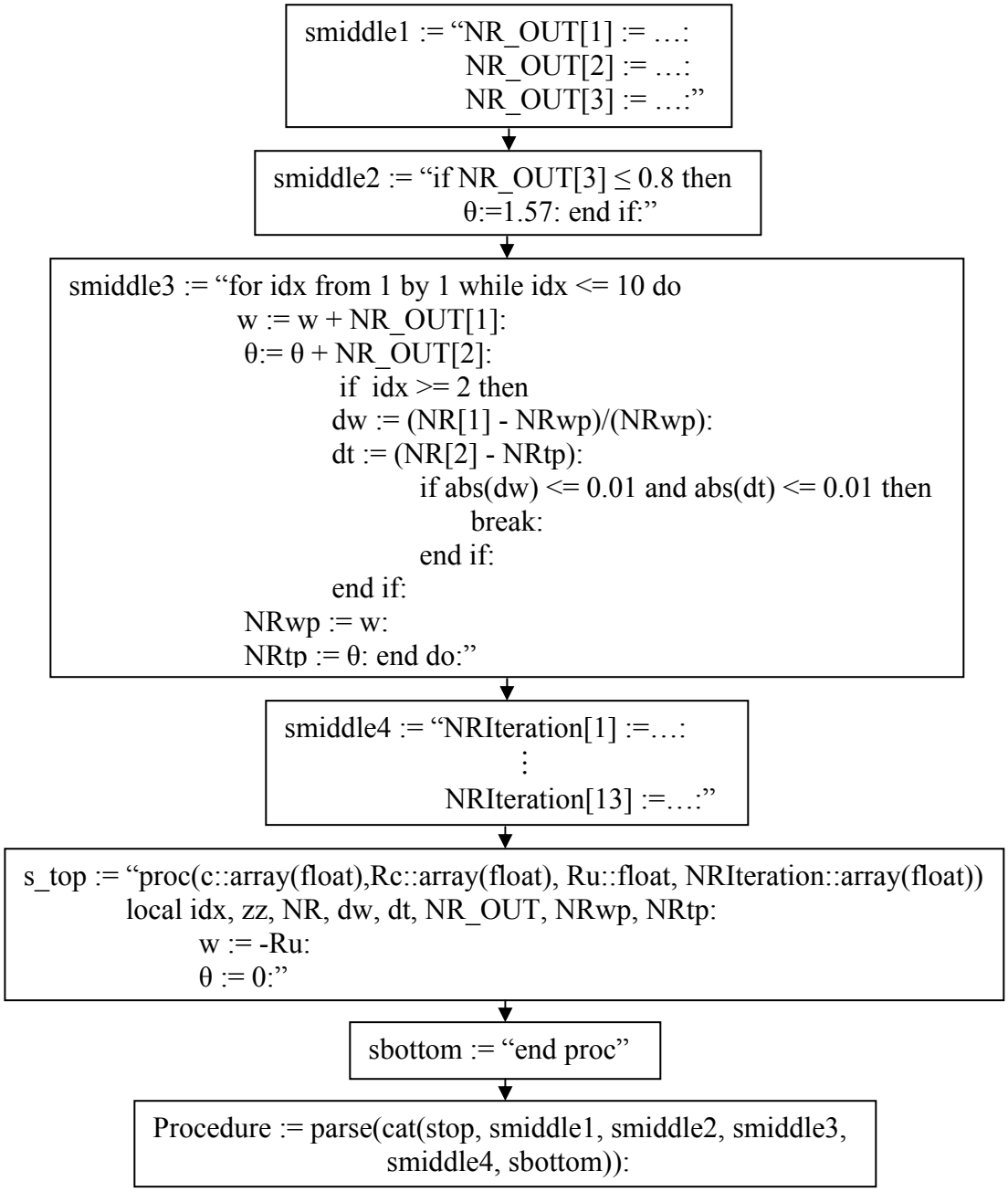


Figure 4.6: The Steps Necessary to Automatically Construct the Road Model Procedure

x, y, R, and Z as input and performs a few operations before returning the value of d. The original procedure is inefficient in determining the value of d since it requires 10 more operations than the optimized procedure to obtain the final result. In the original procedure there are several repeated $\sin(x)$ and $\cos(x)$ terms resulting in a large number of function evaluations. Each time $\sin(x)$ or $\cos(x)$ needs to be evaluated an external function is used to perform the evaluation. Also, in the

$y_1 = 2\,(\sin(x))\,(\cos(x))$ $\text{simplify}(y_1) = 2\,(\sin(x))\,(\cos(x))$ $\text{combine}(y_1) = \sin(2x)$
$y_2 = \cos^2(x) - 1 - \cos^4(x) + 2\sin^5(x) + \cos^2(x)$ $\text{simplify}(y_2) = \sin^4(x)\,(2\sin(x) - 1)$ $\text{combine}(y_2) =$ $\left(\frac{1}{2}\right)\cos(2x) - \frac{3}{8} - \left(\frac{1}{8}\right)\cos(4x) + \left(\frac{1}{8}\right)\sin(5x) - \left(\frac{5}{8}\right)\sin(3x) + \left(\frac{5}{4}\right)\sin(x)$

Table 4.4: Examples of Maple's *combine()* and *simplify()* Commands

Procedure Before Optimization
8 functions, 3 divisions, 4 additions, and 5 multiplications
proc(x, y, R, Z) local a, b, c, d: $a := \frac{\cos(x)\sin(x)}{sin^2(x)}$: $b := \ln\left(\frac{\cos^2(x)+R\sin(x)}{\cos(x)}\right)$: $c := \frac{a^2}{\tan(x)} + b^2 + \tan(y)$: $d := (a)(b) + Z$: return d: end proc:
Procedure After Optimization
3 functions, 2 divisions, 2 additions, and 3 multiplications
proc(x, y, R, Z) local a, b, c, d, t_1, t_2, t_4: $t_1 := \sin(x)$: $t_2 := \cos(x)$: $a := \frac{t_1}{t_2}$: $t_4 := t_1^2$: $b := \ln\left(\frac{t_4+Rt_2}{t_1}\right)$: return $(b)(a) + Z$: end proc:

Table 4.5: Example of Maple's CodeGeneration Package

original procedure the calculation of c serves no purpose in the evaluation of d and thus is a waste of computational power. In the optimized procedure, the evaluation of any expressions that serve no purpose to calculate the output d are dropped. Temporary variables (the t variables) are used for terms that appear more than once in the expressions to determine the output d. Thus, the CodeGeneration package can be used to obtain an optimized computational sequence within a procedure to improve the computational cost of the procedure.

Optimization on the expressions and procedures can be performed at two different levels. They can be performed at the system level or at the component level. System level optimization is performed on the system equations as a whole after they are formulated and component level optimization is performed on the expressions or procedures as they are created. Either the *simplify()* or *combine()* Maple routines are used to optimize the system equations after they are formulated and the CodeGeneration package is used to optimize the road model procedure as it is formulated. Thus, both types of optimization are used to improve the computational cost of simulating the vehicle rolling over a 3-D road.

During the formulation stage after the equations Δw, $\Delta\theta$, and $detJ$ are formed (the NR_OUTs in Figure (4.4)) and the equations for the outputs from the procedure are formed (the NRIterations in Figure (4.5)), the CodeGeneration package is used to optimize the evaluation of the equations by creating an optimized computational sequence for evaluating the expressions. The procedure is then automatically created around the optimized computational sequence instead of around the NR_OUTs and NRIterations. This is accomplished by taking the column vector of NR_OUTs and combining it with the column vector of NRIterations and then using the CodeGeneration package to optimize the single column vector of equation evaluations — refer to Figure (4.7). The CodeGeneration package will return a computational sequence that includes the evaluation of temporary variables along with the evaluation of the NR_OUTs and the NRIterations. The road model procedure can then be automatically created by following the steps outlined in Section (4.3), with the addition of the declaration of the temporary variables and an additional component at the beginning of the middle section that evaluates the temporary variables. Note, the temporary variables need to also be evaluated within the Newton-Raphson iteration and before the NRIterations are calculated. The end result is an optimized road model procedure that can be automatically created during the formulation stage based on the road profile. Note that for the case when the road is specified in tabular form, the optimized computational sequence to evaluate the expressions (NR_OUTs and NRIterations) can be hard-coded into the road model procedure.

4.5 Structure of Simulation Code

A method to determine the contact point between the tire and the ground has been developed and optimized for simulation. The method is based on the thin disk tire model with variable radius by solving two equations based on the current position and orientation of the tire. The road model procedure, which is built around the Newton-Rapshon iteration, will first determine the contact point and then based on the result, will determine the ISO unit vectors, the vector from the tire center to the point of contact, and the loaded radius. This information is necessary to calculate the intermediate variables that are used by the tire model to calculate the tire forces and moments that are reacted on the vehicle at the wheel center.

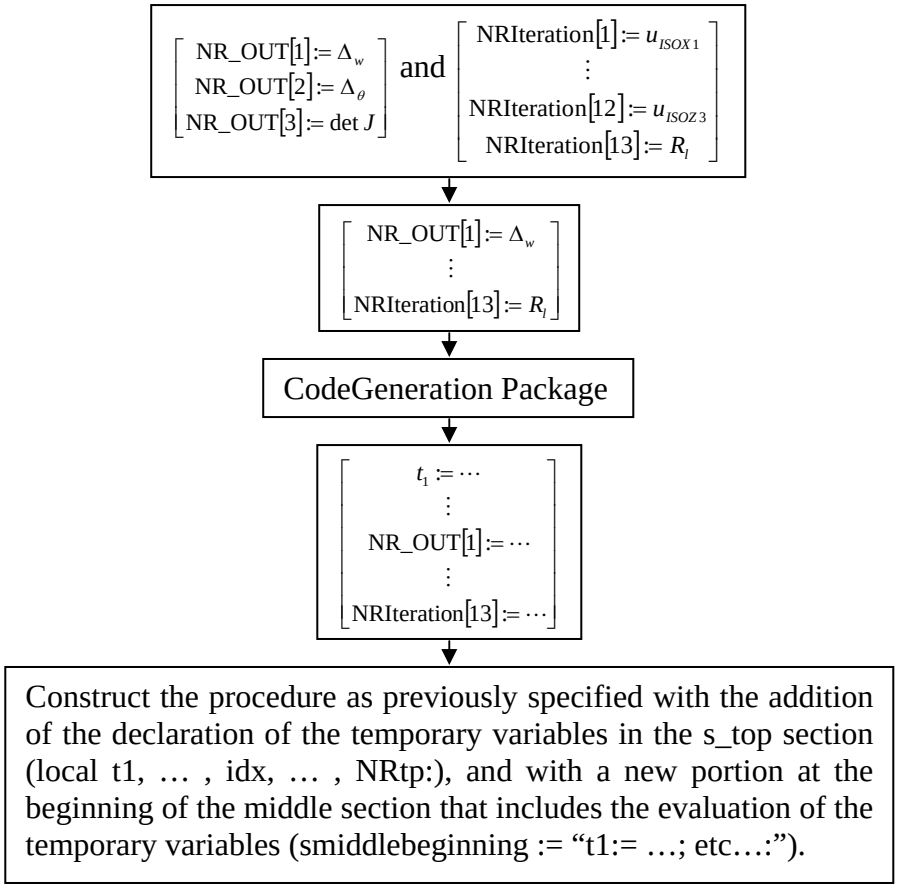


Figure 4.7: The Steps Necessary to Automatically Construct the Road Model Procedure with Simulation Code Optimization

The external forces applied to the wheel center, F_C and M_C, will be part of the generalized force matrix and thus need to be calculated during simulation to determine the response of the vehicle. These forces are dependent on the tire forces and moments that are calculated from the tire model which are dependent on a set of tire intermediate variables. The direction of the tire forces and moments (the ISO unit vectors) and the tire intermediate variables are dependent on the results from the road model procedure. Therefore, there is a need to determine a systematic manner to evaluate F_C and M_C during simulation. One of the possible solutions is to determine an expression for the forces using symbolics by complete substitution during the formulation stage. However, in most cases, a single expression does not exist because most tire models contain if statements, such as the Fiala tire model, to determine the forces and moments which require values that are dependent on the current conditions of the vehicle. Also, the expressions for the equations can become quite large depending on the road profile and thus can take a long time to formulate or may be too large to evaluate. Therefore, a method based on a structured sequence (a sequence that evaluates the forces in steps) needs to be

created to evaluate the forces and moments during simulation.

Morency [70] developed a method to formulate the governing equations using a symbolic approach that includes the evaluation of functions. For vehicle systems rolling on a flat road, Morency [70] developed a systematic fashion to determine the tire intermediate variables and to call functions during simulation to evaluate the tire forces and moments, and to determine the longitudinal slip and lateral slip angle when transients are important. Essentially, dummy variables are used to represent F_C and M_C during the formulation stage and a computational sequence is created to evaluate them during simulation by performing a number of expression and function evaluations [70]. The end result is a simulation code structure that permits F_C and M_C to be evaluated during simulation allowing the generalized force matrix to be updated.

The simulation code structure developed by Morency can be extended to include an additional function call to the road model procedure. This is accomplished by updating the computational sequence created by Morency, to first perform a function call to determine the location of the contact point between the tire and the ground; refer to Figure (4.8). Essentially, during simulation when it comes time

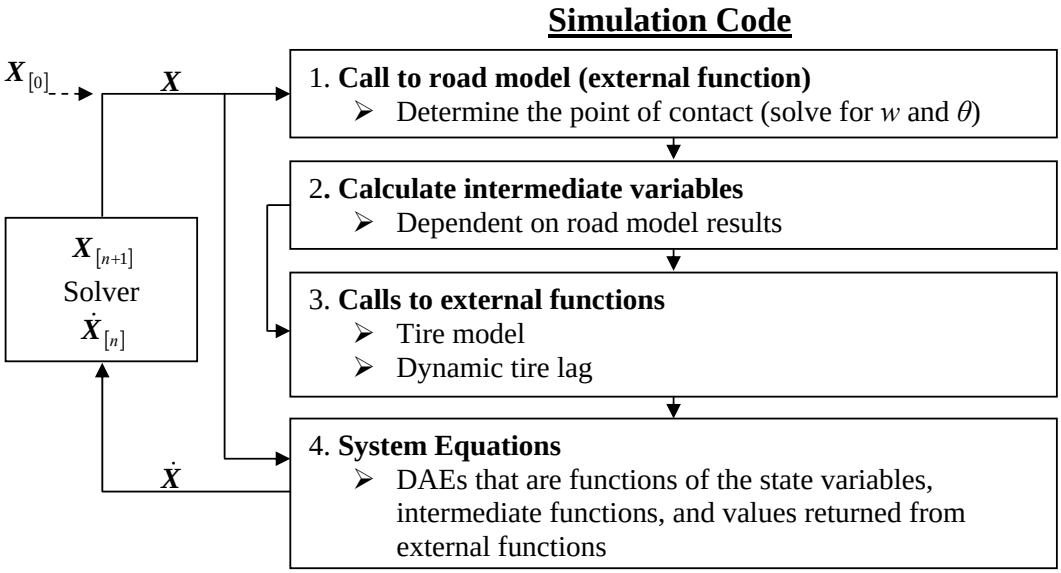


Figure 4.8: Structure of Simulation Code

to evaluate F_C and M_C, the road model procedure is first called to determine the ISO unit vectors, $\vec{r}_{P/C}$, and R_l that are then used to calculate the tire intermediate variables. These variables are then sent to a tire model to calculate the tire forces and moments which are then reacted at the wheel center through Equations (3.1) and (3.2). This simulation code structure is applied to every tire so that the simulation can move forward by one time step. Note, an additional function can be called to determine the derivatives of extra state variables to model the tire transient behavior (Equations (3.23) and (3.24)).

The first process (or block) in the simulation code structure is to use the road model procedure defined in Sections (4.3) and (4.4) to determine the contact point. The second block involves the calculation of the tire intermediate variables. The list of variables that are used in the present work is shown in Table (4.6). It is to

Intermediate Variable	Symbolic Expression	Order of Evaluation			
γ	$\arcsin((\hat{u}_{ISOY} \times \hat{u}_{SymAxis}) \cdot \hat{u}_{ISOX})$	■			
$\dot{\delta}$	$\vec{V}_C \cdot \hat{u}_{ISOZ}$	■			
R_{eff}	R_u or R_l or Equation (3.16)	■			
Ω	$[\vec{\omega}_C \cdot \hat{u}_{SymAxis} - (\vec{\omega}_C \cdot \hat{u}_{ISOZ}) \sin(\gamma)] / \cos^2(\gamma)$		■		
F_Z	$max((k_t\delta + c_t\dot{\delta}), 0)$		■	■	■
$\mathbf{r}_{E/C}$	$(R_{eff}/R_l) \cdot \vec{r}_{P/C}$		■		
V_{XE}	$\left(\vec{V}_C + \vec{\omega}_C \times \vec{r}_{E/C}\right) \cdot \hat{u}_{ISOX}$			■	
V^g_{XE}	$\left(\vec{V}_C + (\vec{\omega}_C - \Omega\hat{u}_{SymAxis}) \times \vec{r}_{E/C}\right) \cdot \hat{u}_{ISOX}$			■	
S	$-V_{XE}/\lvert V^g_{XE}\rvert$				■
V^g_{XP}	$\left(\vec{V}_C + (\vec{\omega}_C - \Omega\hat{u}_{SymAxis}) \times \vec{r}_{P/C}\right) \cdot \hat{u}_{ISOX}$			■	
V^g_{YP}	$\left(\vec{V}_C + (\vec{\omega}_C - \Omega\hat{u}_{SymAxis}) \times \vec{r}_{P/C}\right) \cdot \hat{u}_{ISOY}$			■	
α	$\arctan\left(V^g_{YP} \lvert V^g_{XP}\rvert\right)$				■

Table 4.6: The Logical Order of Evaluations to Calculate the Tire Intermediate Variables

be noted that this list of variables is sufficient for the tire models presented in this thesis. However, this list may not be adequate for other tire models that the user may wish to include, but will work with the majority of models. In the calculation of the tire intermediate variables, the user has the option to choose which method is used to calculate the effective rolling radius and whether or not tire transients are important.

It can be seen from the list of tire intermediate variables that some of them are dependent on others and thus need to be evaluated before the others. The variables at the end of the list are functions of the ones at the beginning of the list. It is logical that the variables at the beginning need to be evaluated before the ones at the end as can be seen in the pseudo-Gantt chart shown in Table (4.6). Thus, in the second block of the simulation code structure, the tire intermediate variables are evaluated in the logical manner presented in Table (4.6).

There are two methods that can be used to calculate the tire intermediate variables and these are by direct substitution or by using a computational sequence. The direct substitution method will follow the logical manner to determine a symbolic expression for each of the tire intermediate variables required by the tire model. In certain cases, it may not be possible to determine a single expression for some of the intermediate variables. Also, the expressions may contain several

function evaluations and operations that require a large amount of computational power to perform. This was shown by Morency [70], as he determined the computational power associated with evaluating each of the expressions symbolically for the case when the tire is rolling on a flat surface. Thus, it is best to use a computational sequence to evaluate the tire intermediate variables in the logical fashion. A computational sequence will evaluate each of the expressions and store them temporally so that they can be used to evaluate other tire intermediate variables, and by the tire model. Doing this is similar to performing an optimization on the evaluation of the tire intermediate variables, as was shown in Section (4.4).

The third block of the simulation code structure involves calls to external functions that are necessary to return values required to evaluate expressions in the system equations, so that they can be solved. An external function is used to calculate a variable that is not a state variable and appears in the system equations but can not be fully evaluated during simulation. The majority of tire models fit this criteria because the tire forces and moments appear in the system equations, but can not be evaluated until simulation because they contain "if" statements that can only be evaluated during simulation (Pacejka tire model for example). External functions can also be used to return the derivatives of extra state variables (Equations (3.23) and (3.24) for example) when there is "if" constructs that are required in the calculation of the derivatives of the extra states (Table (3.1) for example). In either case, each external function will take in a list of variables (tire intermediate variables for example) and will return the values in array form (the tire forces and moments for example).

In the present case, there are only two external function calls needed at this stage, and these are the tire model and the function necessary to determine the relaxation lengths when tire transients are important. It is to be noted that the use of external functions also allows the user to create his/her own tire model to evaluate the tire forces and moments. The user also has the option to create his/her own road model to determine the contact point.

The final portion of the simulation code structure is to evaluate F_C and M_C so that the generalized force matrix can be updated. The simulation code structure is applied to every tire in the vehicle model so that the generalized force matrix can be updated. Therefore, a symbolic naming convention needs to be implemented so that each tire can go through the simulation code structure independently.

The naming convention described by Morency [70] is used in the present work. During the first portion of the computational sequence, the inputs to the road model and the outputs from the model are given distinctive names that relate to the name of the tire. It is not the function that changes between each tire, it's the inputs and outputs to and from the function that change. Similarly, during the third portion of the computational sequence, the inputs to and from the tire model and the tire transient function (if tire transients are important) are given distinctive names. Also, during the second portion, the tire intermediate variables are constructed so that during simulation they are assigned to a set of dummy

variables that are named (and thus dependent on) according to the name of the tire. Thus, the forces and moments between each tire and the road can be evaluated independently following this name convention.

It is instructive to give an example involving the simulation code structure. Consider a 2-wheeled vehicle where both the front and rear tires are modeled using the Fiala tire model. Suppose that based on experimental results it is known that the tire transients are only important on the front wheel. Let the front tire be named "Front" and the rear tire to be named "Rear". The simulation code structure for this example is shown in Table (4.7). It is to be noted that the extra differential equations necessary to obtain the tire forces and moments are appended to the set of differential equations that represent the governing equations. Thus, in the example shown, the two differential equations necessary to represent the tire transient behavior (Equations (3.23) and (3.24)) are appended to the ODEs that represent the system. Also, the road model function is named RoadModel, and the function necessary to determine the tire transients based on the equations defined in Section (3.2.5) is named TransientFunc. As well as, $q_{1dot} = dS/dt$ and $q_{2dot} = d(tan(\alpha))/dt$. Therefore, the simulation code structure presented above can be used to analyze wheel vehicle systems rolling over 3-D roads when the governing equations are formulated using a symbolic formulation approach.

The structure of simulation code outlined above assumes that the road is modeled as a single elevation function ($z = f(x, y)$). However, as outlined in Section (3.5), there are situations where the road can not be modeled as a single elevation function and is thus modeled as a piecewise cubic spline (or cubic spline) through a set of tabular data that represents the centerline of the road. There is an additional step required in the simulation code structure when the road is modeled as a spline, and this step is the determination of the spline coefficients that represent the road at the current time step. The updated simulation code structure is shown in Figure (4.9). It is to be noted that during the formulation stage, the set of tabular data that represents the road is first used to determine the cubic spline coefficients (AS, BS, CS, and DS) between all sets of data points (Equations (3.59)-(3.62)) and to determine the distance traveled along the path (Equation (3.57)). Once the spline coefficients and the distance are determined, they are stored into an array that is used during simulation to apply the appropriate coefficients to the function that represents the road (Equation (3.66)). Thus, the first portion of the simulation code structure becomes an external function call to a procedure that determines where the tire is located along the centerline of the road, and based on its position determines which coefficients to use to represent the road. The remainder of the simulation code structure is the same as the one shown in Figure (4.8). Note also that the naming convention outlined above will also be used in the updated simulation code structure.

When the centerline of the road is specified, the center of the vehicle should follow the path defined by the centerline of the road and a driver model should be used to enforce this [23]. The current position of the tire is taken as the position of the axle of the tire along the centerline of the vehicle (refer to Figure (3.25)). It

First Portion

$c_{Front} := \cdots$

$\vdots$

$R_{u\_Rear} := \cdots$

$\text{NRIteration}_{Front} := \text{RoadModel}(c_{Front}[1], \cdots, R_{u\_Front})$

$\text{NRIteration}_{Rear} := \text{RoadModel}(c_{Rear}[1], \cdots, R_{u\_Rear})$

$u_{ISOX1\_Front} := \text{NRIteration}_{Front}[1]$

$\vdots$

$R_{l\_Front} := \text{NRIteration}_{Front}[13]$

$\vdots$

$R_{l\_Rear} := \text{NRIteration}_{Rear}[13]$

Second Portion

$\vdots$

$S_{Front} := (-V_{XE\_Front})/(|V^g_{XE\_Front}|)$

$\vdots$

$\alpha_{Rear} := (-V^g_{XP\_Front})/(|V^g_{YP\_Front}|)$

Third Portion

$TireModelParams_{Front}[1] := 0.16$

$\vdots$

$TireModelParams_{Front}[6] := 0.2$

$ForceValues_{Front} := \text{Fiala}(F_{Z\_Front}, \cdots, TireModelParams_{Front})$

$F_{x\_Front} := \text{ForceValues}_{Front}[1]$

$\vdots$

$M_{z\_Front} := \text{ForceValues}_{Front}[6]$

$TireModelParams_{Rear}[1] := 0.155$

$\vdots$

$M_{z\_Rear} := \text{ForceValues}_{Rear}[6]$

$SlipStateParams_{Front}[1] := 5900$

$\vdots$

$SlipDerivatives_{Front} := \text{TransientFunc}(F_{z\_Front}, \cdots, SlipStateParams_{Front})$

$q_{1dot\_Front} := SlipDerivatives_{Front}[1]$

$q_{2dot\_Front} := SlipDerivatives_{Front}[2]$

Fourth Portion

$\vdots$

$\dot{x}_{[n]} := q_{1dot\_Front}$

$\dot{x}_{[n+1]} := q_{2dot\_Front}$

Table 4.7: Example of the Simulation Code Structure

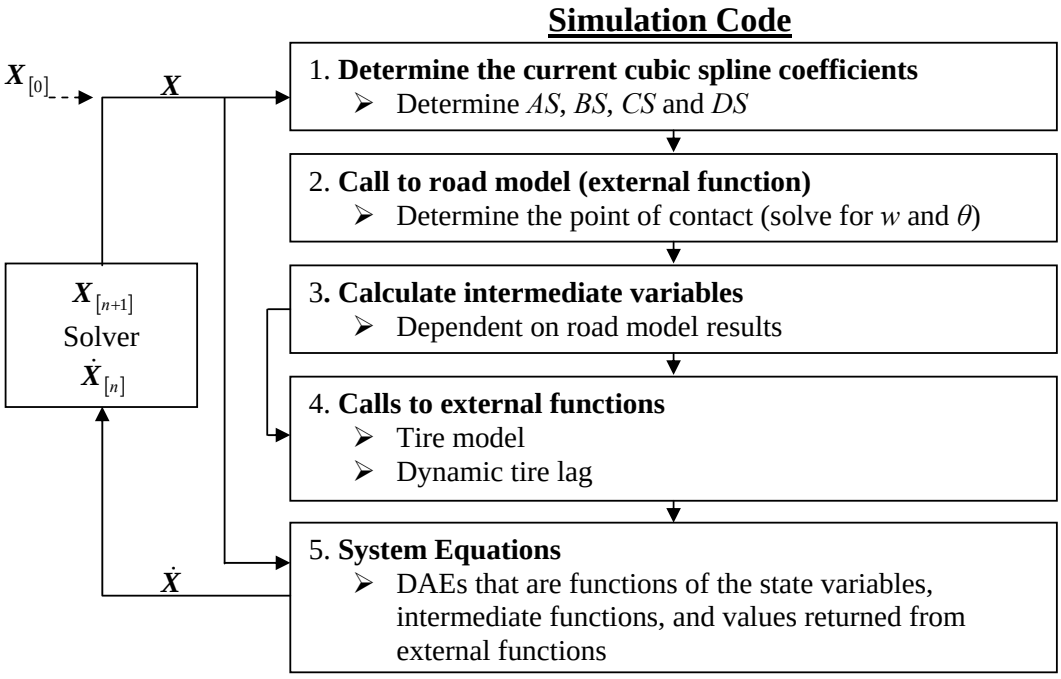


Figure 4.9: Structure of Simulation Code when the Road is Modeled as a Set of Tabular Data

is used as the input to the procedure to locate the tire in the table, along with the array of the spline coefficients and the set of tabular data that includes the distance traveled along the path, s_{table}. There are several different methods that can be used to search through the set of tabular data based on a single scalar variable. Note that it is easy to search through the table because there are no two data points that will have the same distance traveled along the path (s_{table}). The bisection method is used in the present work because it is a simplified method, and since the method will converge to the result at an exponential rate [77].

The underlying concept behind the bisection method is simple. Over the interval of the table, it is known that at some point, Δs will change sign. Δs is the difference between the current distance traveled by the tires axle along the centerline of the road, and the distance traveled along the centerline of the road that is recorded in the table at the data point of interest ($s - s_{table}$). Δs is first evaluated at the limits of the interval in consideration and then at the midpoint of the interval. The value of Δs at the midpoint of the interval is used to replace whichever limit has the same sign as the value of Δs at the midpoint [77]. This process is continued until the desired error of convergence is achieved. The bisection method that is used in this procedure is shown in Table (4.8). Note, that *dim* is the number of entries in the table, or the size of the array that represents s_{table}. The output of this procedure is the current value of each of the spline coefficients, along with β (the angle that locates the y_{road} axis) and the bank angle. It is to be noted that for the case when the road only varies with respect to the global X axis and is represented as a cubic

```
L := 1 :
R := dim :
for i from 1 by 1 while i <= 40 do
  p := trunc((R − L)(1/2)) :
  p := L + p :
  if s_table[p] > s then
    R := p :
  end if:
  if s_table[p] < s then
    L := p :
  end if:
  if i >= 2 then
    dp := abs((p − pold)/pold) :
    if dp<=0.1 then
      break:
    end if:
  end if:
  pold := p :
end do:
```

Table 4.8: The Bisection Method Used to Locate the Current Position of the Tire in the Set of Data Points that Represent the Road

spline in terms of x, the bisection method will still be used to determine the current location of the tire. However, the current x position of the tire will be the variable used as input to the procedure and the position in the table determined using Δx instead of Δs.

The actual distance traveled along the path of the centerline of the road is the total distance that the vehicle has traveled. This distance can be obtained by integrating the differential equation shown in Equation (4.7).

$$\frac{ds}{dt} = V_x(t) \tag{4.7}$$

Note, that $V_x(t)$ is the velocity of the vehicle with respect to its local x axis which is always pointing in the direction of vehicle heading [43]. $V_x(t)$ can also be determined from the state variables that represent the system. Thus, when the road is specified in tabular form, Equation (4.7) will be appended to the set of ODEs that represent the system so that the distance traveled along the path by the vehicle can be determined. Note that this distance will give the distance traveled along the path by the center of mass of the vehicle. Thus, the relative position of the axle of the tire will have to be added to this distance so that the proper distance is used to locate the current cubic spline coefficients; refer to Equation (3.68).

4.6 Integration with DynaFlexPro

The simulation code structure needs to be implemented in conjunction with a method that formulates the governing equations using a symbolic approach so that the structure can be justified. It can be implemented in a program that uses any form to formulate the governing equations symbolically. However, linear graph theory will be used to formulate the governing equations since the topology is not fixed, can analyze multi-domain systems, and offers the choice of the modelling variables trough the selection of the spanning tree [67]. The structure is implemented in the DynaFlexPro (DFP) software because it formulates the equations using linear graph theory and because it is based on the Maple symbolic language [8].

The linear graph representation of the tire component defined by Schmitke et al. [89] can be extended to include a 3-D road. This is accomplished by extending the third component of the linear graph (F_3) to include the additional step to calculate the contact point between the tire and the ground [24]. The linear graph representation of the tire and the ground is shown in Figure (4.10). The node la-

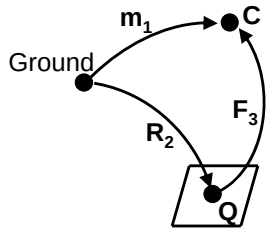


Figure 4.10: Linear Graph Theory Representation of the Tire and the Road

beled Ground represents the inertial reference frame, the node labeled Q represents the ground reference frame, and the node labeled C defines a frame fixed on the tire that is located at its center. Note, Q is assumed to be equivalent to the global reference frame. This linear graph representation of the tire and the ground will define the tire/road component in the mechanical translational domain as well as in the mechanical rotational domain.

Edge m_1 represents the mass, weight in a local gravitational field, and inertia of the tire. Edge F_3 represents the forces and moments applied to the center of the tire. A set of constitutive equations is used to represent the physical behavior of each edge in the linear graph [65]. The equations that represent the physical behavior of edge m_1 are the Newton-Euler equations for a rigid body expressed in D'Alembert form, and the equations that represent edge F_3 are Equation (3.1) and (3.2). These terminal equations are shown in Table (4.8). Note, m is the mass of the tire, $\mathbf{J}$ is the inertia dyadic of the tire, and $\vec{g}$ is the local gravitational field. It is to be noted that the rigid body assumption is only valid if the deformation of the tire does not significantly affect its center of mass location or its rotational inertia. However, it can be noted, that this is a reasonable assumption when the vehicle model is used to analyze the vehicle in handling situations [23].

Edge	Terminal equation in the mechanical domain	Terminal equation in the rotational domain
m_1	$\vec{F}_{m1} = -m\dot{\vec{v}}_{m1} + m\vec{g}$	$\vec{M}_{m1} = -\mathbf{J}\dot{\vec{\omega}}_{m1} - \vec{\omega}_{m1} \times (\mathbf{J}\vec{\omega}_{m1})$
R_2	$\vec{F}_{R2} = \vec{0}$	$\vec{M}_{R2} = \vec{r}_{Q/C} \times \vec{F}_P$
F_3	$\vec{F}_{F3} = \vec{F}_P$	$\vec{M}_{F3} = \vec{M}_P + \vec{r}_{P/C} \times \vec{F}_P$

Table 4.9: The Terminal Equations of the Tire/Road Component

To properly represent the physics of the interaction between the tire and the road, the moment applied at the road reference frame must be different than the moment applied to the wheel center [23]. It is important to note that each edge in a linear graph can only have one through variable associated with it per derivative level [65]. Thus, edge F_3 can not describe both the moment applied to node C and node Q. Edge F_3 describes the moment applied to node C only. The moment representing edge R_2 is applied to the edge to ensure that the proper moment is assigned to node Q. Consider the algebraic sum of the moments applied to node Q by the tire/road component.

$$\begin{aligned}
\sum \vec{M}_{\text{applied to node Q}} &= \vec{M}_{R2} - \vec{M}_{F3} && (4.8)\\
&= \vec{r}_{Q/C} \times \vec{F}_P - \left(\vec{M}_P + \vec{r}_{P/C} \times \vec{F}_P\right) && (4.9)\\
&= -\vec{M}_P + \left(\vec{r}_{P/C} - \vec{r}_{Q/C}\right) \times -\vec{F}_P && (4.10)\\
&= -\vec{M}_P + \vec{r}_{P/Q} \times -\vec{F}_P && (4.11)
\end{aligned}$$

The sum shows that the moment representing edge R_2 is necessary to ensure that the proper moment is applied to node Q so that the forces and moments are balanced.

The tire forces and moments ($\vec{F}_P$ and $\vec{M}_P$) will be resolved into the ISO tire axis system, and their magnitudes given by a tire model. These variables will appear in the generalized force matrix of the system equations through edge F_3 of the tire/road component. The other variables in the terminal equations are $\vec{r}_{P/C}$ which is obtained from the road model, and $\vec{r}_{Q/C}$ which is equal to $-\vec{c}$ because Q is equivalent to the global reference frame. These variables will appear in the system equations, but need to wait until simulation to be evaluated. Therefore, when the governing equations are formulated, these variables are left as dummy variables that are assigned during simulation by following the simulation code structure.

The method necessary to solve these dummy variables is to follow the simulation code structure using the component level templates created by Morency [70]. There are three levels of information and these are as follows: topological information, edge level information, and component level information [88]. The topological information consists of the connectivity between the edges and the nodes that describe the system and the information is necessary to construct the system equations. The edge level information consists of the information associated

with describing the behavior of the edge. Such information increases the terminal equation and the through and across variables of the edge. The component level information is any information that can not be associated with an individual edge [88].

Component level information was first introduced by Schmitke and McPhee [88] to analyze mechatronic systems by creating linear graph components that represent subsystems of large mechatronic devices. Each subsystem was named an equivalent subsystem component (ESC). An ESC usually involves a few nodes where the subsystem connects to its surroundings and a minimum number of edges that serve to connect the boundary nodes. For each ESC, the dynamics are described by internal variables and equations that were stored as component level information [88, 87]. Using ESCs offers the advantage that larger systems can be broken down into components and that the subsystem models can be retrieved from a library of ESCs resulting in a reduced time to formulate the governing equations. The component level algebraic equations from the ESCs are appended to the set of algebraic equations used to represent the system equations [88]. Similarly, the component level differential equations from the ESCs are appended to the governing equations that are used to represent the system [88].

Morency [70] developed an ESC to represent the tire on a flat surface. He extended the work done by Schmitke and McPhee to include the evaluation of intermediate variables and function calls in an ESC. Essentially, Morency [70] created the ESC that represents the tire component as a simulation code structure. He permitted the tire to be pre-defined as a component that permits the simulation code structure to be used to evaluate the dummy variables in the system equations during simulation. Thus, the structure of simulation code defined in Section (4.5) is used to define the ESC that represents the tire/road component. Based on this ESC, DynaFlexPro can be used to analyze wheeled vehicle systems rolling on 3-D roads.

Chapter 5

Examples

5.1 Overview

The tire/road component developed in Chapter 4 using the underlying theory presented in Chapter 3 is programmed in the Maple symbolic language and implemented in DynaFlexPro. This component permits the tire forces and moments to be evaluated when the vehicle is rolling on a 3-D road. DynaFlexPro is used to formulate the governing equations that describe the behavior of the Chevrolet Equinox in different situations. Each example consists of a different road profile with the vehicle performing either a braking maneuver, turning maneuver, or both.

The tire/road component is validated by comparing the different examples to a well established tool for vehicle dynamic studies (MSC.ADAMS). MSC.ADAMS is an industry standard for analyzing the complex behavior of mechanical systems including the vehicle [23]. The example models permit the theory presented in Chapter 3 to be evaluated. They also show the versatility of modelling wheeled vehicle systems on 3-D roads using a symbolic modelling approach.

The model of the Chevrolet Equinox is presented first, followed by a discussion of the modelling aspects used to simulate the vehicle in both MSC.ADAMS and DynaFlexPro. Examples based on roads described by a single elevation function are presented, followed by examples based on roads that need to be defined in tabular form. Each example includes a discussion of the results.

5.2 Vehicle Model

The Chevrolet Equinox is a sport utility vehicle (SUV) that has proven itself in its vehicle category [14]. It is an all-wheel drive vehicle with an excellent fuel efficiency, and has won the first ever Green Car Vision Award by the Green Car Journal for the most eco-friendly vehicle [34, 58]. The engineers have been making improvements to the emissions from the Equinox and to other environmental factors. Overall, the

Equinox is a comfortable vehicle to drive for every day purposes while still having the capability to perform all of the essentials of a 4 wheel drive vehicle.

The vehicle model defining the 2005 Chevrolet Equinox is the 14 degree of freedom (DOF) model proposed by Michael Sayers [82], which was discussed in Section (2.3). The tire is included in the vehicle model using the tire/road component outlined in Chapter (4). Thus, the road is incorporated in the model through the tire/road component.

5.3 Modelling Aspects

The first step to creating the model is to represent it as a linear graph. This is accomplished using the ModelBuilder graphical user interface that is embedded into DynaFlexPro. ModelBuilder allows the user to describe the topology of the system and to specify the spanning tree that represents the system in both the translational and rotational domains [8]. DynaFlexPro will automatically assemble a linear graph representation of the system model based on the ModelBuilder description. Details of ModelBuilder can be found in the DynaFlexPro user manual [8]. Figure (5.1) shows the vehicle model that was created in ModelBuilder to represent the topology of the Chevrolet Equinox. The tire (FLtire for example) in Figure (5.1) represents

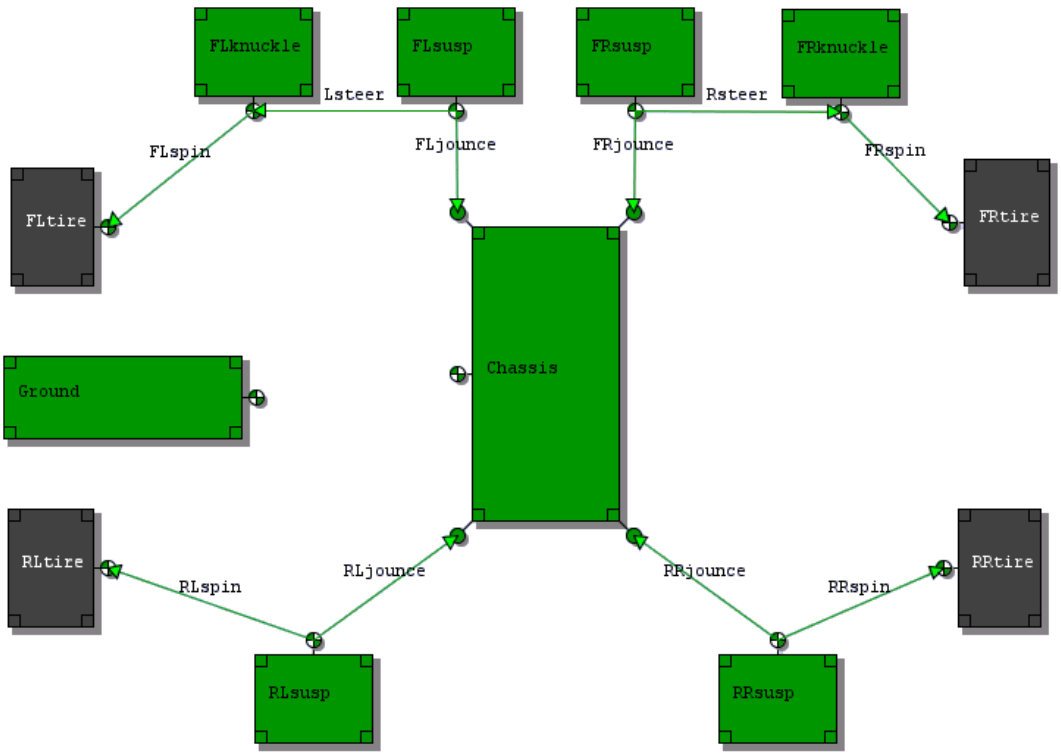


Figure 5.1: The Vehicle Model Created in ModelBuilder

the linear graph shown in Figure (4.10). The spin edges are the revolute joints that permit the tire to rotate about the spin axis (symmetry axis). Note, each arrow in Figure (5.1) represents an edge which defines a joint. The steer edges are motion drivers where the steering motion with respect to the Z axis of the vehicle is prescribed. The jounce edges are the prismatic joints that represent the suspension system. Note, each prismatic joint has a stiffness and damping associated with it. The boxes/rectangles (Chassis for example) represent the rigid bodies in the model. Note, each tire component does have a rigid body component associated with it.

A linear graph representation of the Chevrolet Equinox is shown in Figure (5.2) for the translational domain. Note, ModelBuilder permits the user to view the linear

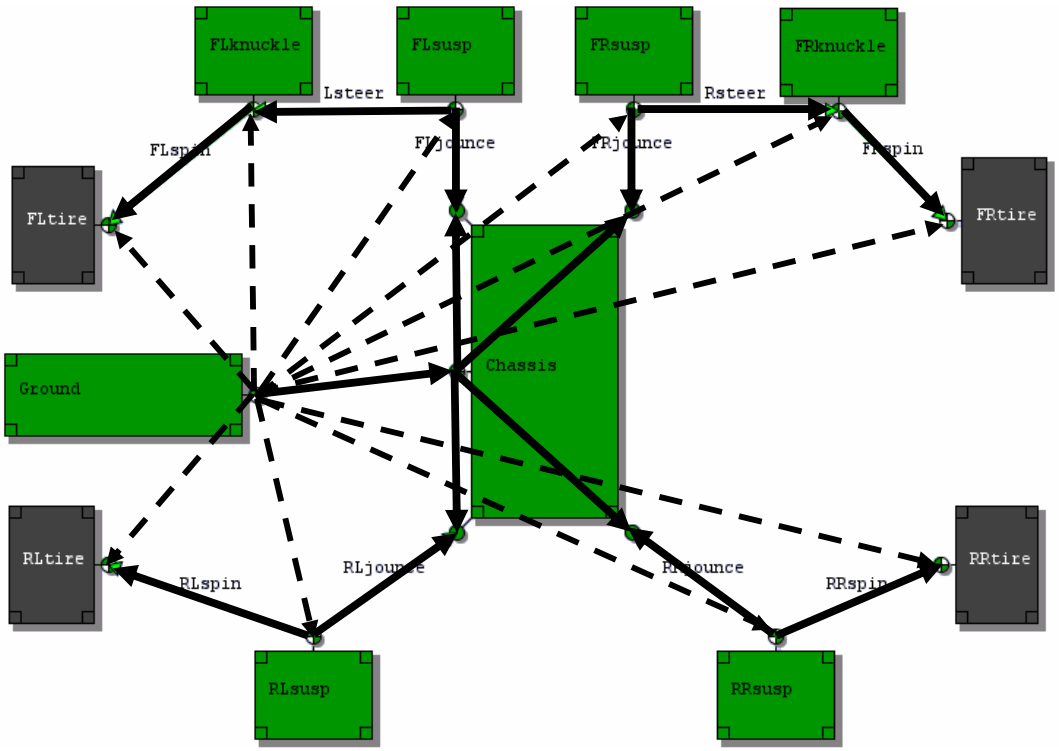


Figure 5.2: Linear Graph Representation of the Vehicle Model

graph of the system in both the translational and rotational domains. The linear graph is shown on top of the ModelBuilder description for clarification purposes. The linear graph representing the tire is not shown in the figure to eliminate it from being too clustered. Instead, a detailed linear graph for one corner of the vehicle model is shown in Figure (5.3) since the graph is identical in every corner with the exception that the front corners contain an additional edge representing the steering motion. The tire/road component is shown by the dashed-dot lines. The naming convention used in the linear graph is as follows.

- Body is used to indicate that the edge denotes a rigid body component, and thus represents the inertia and weight of the body in a gravitational field

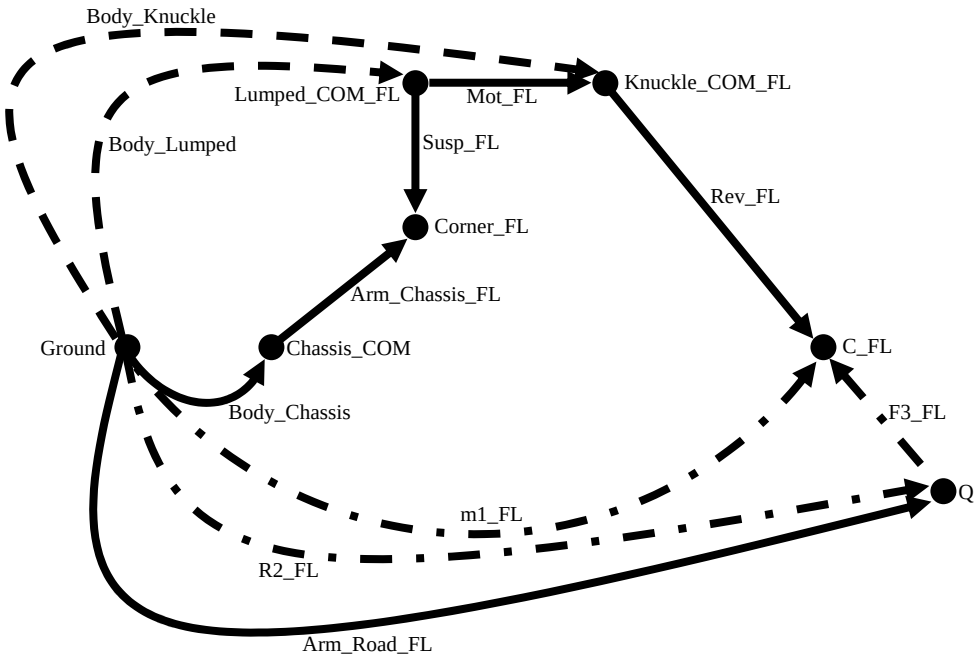


Figure 5.3: Linear Graph Representation of the Front Left Corner of the Vehicle Model

- Arm indicates that the edge is a rigid arm component, and thus describes a translational and rotational transformation between two frames
- Rev shows that the edge is a revolute joint component
- Mot indicates that the edge is a motion driver component, which forces a translation and rotation between two frames
- Susp specifies that the edge is a prismatic joint representing the suspension system and thus has a stiffness and damping associated with it

It can be noted that the rotational graph will be identical to the translational graph. The chassis is the vehicle body, lumped is the unsprung mass of the suspension system, and the corner is the location on the vehicle body where the suspension system connects to it. FL represents that the edges/nodes belong to the front left corner of the vehicle and is a naming convention used to distinguish the components in the vehicle model.

The road reference frame (frame Q) connects to the ground frame through a rigid arm component (Arm_Road_FL) that enforces zero translation and rotation between the two frames. Thus, the road reference frame will remain equal to the ground reference frame. The road is specified as an internal part of the tire/road component and thus the road reference frame can be equivalent to the ground frame. The road is taken into consideration through the calculation of F_p and M_P

that appear in edge F_3 of the tire/road component. The road reference frame in the linear graph is to ensure that there is a moment and force balance.

The governing equations are determined by projecting the fundamental cutset equations onto the across space for the edges selected into the tree [65]. Thus, it is the edges that are selected into the tree (both translational and rotational trees) that determine the number of generalized variables used to construct the governing equations. Therefore, selecting the revolute joints (1 DOF each), prismatic joints (1 DOF each), chassis rigid body (6 DOF), arm elements (no DOF), and the motion drivers (no unknown DOF) into the tree results in 14 generalized coordinates ($4 + 4 + 6 = 14$). This implies that a complete set of ordinary differential equations (ODEs) is used to represent the system and not a set of differential algebraic equations (DAEs) since the number of generalized coordinates is equal to the number of degrees of freedom. The tree selection that gives these 14 ODEs is shown in Figure (5.2) as the solid arrows. The cotree is also shown in the figure and is represented by the dashed arrows. This tree selection will be used to obtain the 14 ODEs by projecting the fundamental cutset equations onto the across space of the edges selected into the tree. Any other tree selection with more than 1 rigid body component will result in a set of DAEs (will have more generalized coordinates than DOF).

A state space representation of the governing equations is used. Thus, a set of 28 first order ODEs is utilized to represent the system. It can be noted that if the road is specified in tabular form by defining the centerline of the road then the 4 ODEs used to track the distance traveled along the path by each of the tires is appended to the set of 28 ODEs that represent the system to give a complete set of 32 ODEs. Note that if tire transients are considered, then the set of ODEs to determine the slip parameters would also be appended to the set of ODEs representing the governing equations.

It can be further noted that the coordinates used to track the position of the vehicle body are expressed in the ground frame and their generalized speeds (both the velocity and angular velocity) are expressed in the local frame of the body (frame at the center of mass (COM) of the body). The 321 Euler angles are used to track the orientation of the vehicle body.

The governing equations are formulated for each example using DynaFlexPro based on the linear graph selection shown in Figure (5.2). The type of expression manipulation performed on the governing equations after they are formulated is Maple's *simplify()* command. Morency [70] has shown that the *simplify()* command works best to simplify the governing equations of vehicle systems.

An MSC.ADAMS model is also created for the vehicle model outlined in Figure (2.2) for validation purposes. The model is created using the graphical user interface in ADAMS/View. The tire is added to the model using the ADAMS/View special force tire component for all 4 tires with each of them referencing the same tire and road data file. The 3D spline road is used to model the road unless otherwise specified. With this road profile, the centerline of the road is specified along

with the road bank angle and the road friction coefficients [3]. MSC.ADAMS uses a numerical approach to formulate the governing equations using absolute coordinates, which results in a large number of equations to solve at each time step. Linear graph theory allows joints and motion drivers to be connected directly in series, but the approach in MSC.ADAMS does not permit them to be connected in series and needs a rigid body with a specified mass and inertia in between the two joints. This is the reason for the additional knuckle component in the model between the revolute and motion driver joints. Note that these knuckles have an associated mass and inertia.

The 4th-5th order Runge-Kutta numerical solver is used to solve the governing equations in both MSC.ADAMS (RKF45) and DynaFlexPro/Maple (rkf45) with a tolerance of 0.001 (both absolute and relative tolerance). Details of the solver are given by Press [77]. However, the equations need to be given a set of initial conditions before they can be solved and the initial conditions are listed in Appendix (C) for each state variable. Note, if an initial condition is different then the one recorded in the list, it will be specified within the description of each example.

The Fiala tire model is used for the examples unless otherwise noted to determine the tire forces and moments with no tire transients considered. The list of parameters used in the Fiala tire model are listed in Appendix (B). Note, if it is desired to increase the accuracy of the simulation then it is advised to use the Pacejka model. However, the computational efficiency may decrease when this model is used because of the complexity of the model. The ISO definition for the longitudinal slip and lateral slip angle will be used to calculate the slip parameters (Equations (3.20) and (3.22)). The effective rolling radius is assumed to be equal to the loaded radius.

A detailed description of how the road is modeled is given for each example, along with the details of the MSC.ADAMS 3D spline road model.

The function of time representing the torque applied to each of the revolute joints (torque(t)) is specified as zero, unless noted for each example. The same applies for the steering motion (steer(t)) imposed on the motion drivers. Thus, the vehicle is set to drift with the driver trying to hold the vehicle in a straight line.

All other parameters (mass and inertia properties of the rigid bodies, spring and damping rates, geometry of the vehicle) are shown in Appendix (C).

The simulations were performed for each example in MSC.ADAMS and DynaFlexPro on a PC with Windows XP operating system, 2.33 GHz Intel Core2 Duo CPU Processor, and 4 GB of RAM. The simulation time for each example is given as the average of 5 runs.

5.4 Roads Represented by a Single Elevation Function

The first set of examples presented are ones where the road can be defined by a single elevation function. A set of three examples are considered and these are as follows: the vehicle executes a braking maneuver on a flat road, the vehicle performs a double lane change as it goes over a hill, and the vehicle does a braking maneuver on a banked road. In each of these cases the road model will automatically be constructed in the formulation stage based on the road profile.

5.4.1 Braking Maneuver on a Flat Road

The first example involves the vehicle performing a braking maneuver on a flat road. The braking torque is applied to the vehicle after it has been traveling for 2s and is applied to the wheels for a total duration of 3s. The braking torque starts at 0Nm and then ramps to 1000Nm after 2s and takes 0.5s to reach 1000Nm (ramps to 1000Nm between $t = 2s$ and $t = 2.5s$). The braking torque is then held constant for 2s (until $t = 4.5s$), and then ramps back down to 0Nm in 0.5s. The braking torque as a function of time is shown in Figure (5.4) and in Equation (5.1).

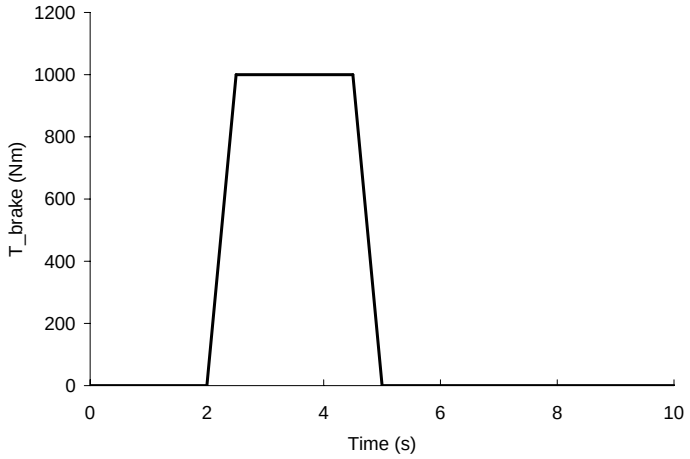


Figure 5.4: The Braking Torque Applied to all Wheels

$$\text{torque(t)} = \begin{cases} & t & < 2s: & 0 \text{ Nm} \\ 2s \leq & t & < 2.5s: & 2000(t-2) \text{ Nm} \\ 2.5s \leq & t & < 4.5s: & 1000 \text{ Nm} \\ 4.5s \leq & t & < 5s: & -2000t + 10000 \text{ Nm} \\ & t & \geq 5s: & 0 \text{ Nm} \end{cases} \tag{5.1}$$

This braking torque can simulate the driver having to slow down the vehicle temporally because of slow traffic ahead or an animal crossing the street. The torque

is modeled in DynaFlexPro using Maple's piecewise function, which will construct the torque function as a piecewise continuous function [12]. The torque is modeled in MSC.ADAMS using the function builder in ADAMS/View by creating a series of if statements.

The road is modeled in DynaFlexPro as the single elevation function that is shown in Equation (5.2).

$$z = f(x, y) = 0 \tag{5.2}$$

The road is created in MSC.ADAMS using the 2D road model with the road type specified as being flat. The road is essentially the XY plane.

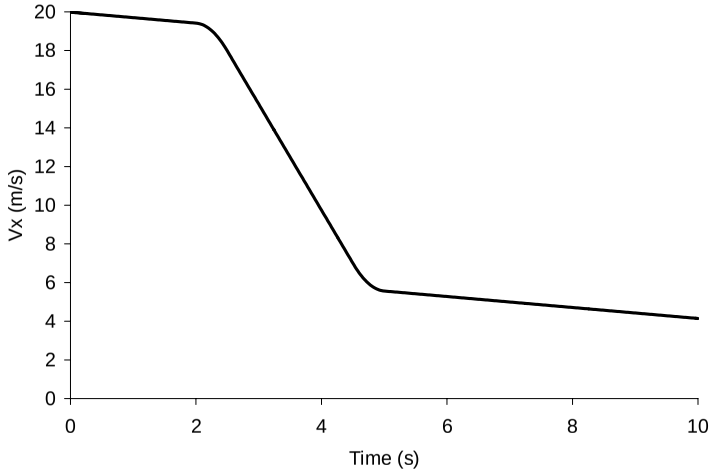
The simulation is performed in both MSC.ADAMS and DynaFlexPro for a duration of 10s. The simulation is also performed in DynaFlexPro using the DynaFlexPro/Tire component (which assumes the road is flat) to compare the simulation times between using the tire/road component to just using the tire component. The results from the example are shown in Figures (5.5) and (5.6) and in Table (5.1).

Software	Simulation Time (s)
DynaFlexPro (with tire/road component)/Maple	13.5
DynaFlexPro (with tire component (default))/Maple	13.6
MSC.ADAMS	164.4

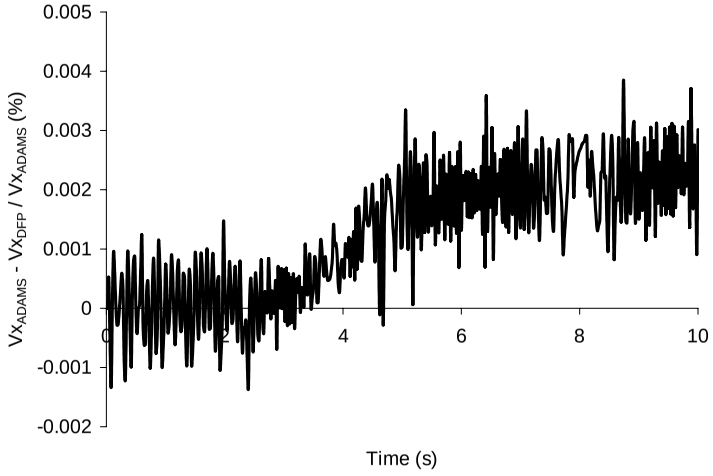
Table 5.1: The Simulation Times for the Example of the Chevrolet Equinox Performing a Braking Maneuver on a Flat Road

It can be seen from the results that when the braking torque is applied to the wheels, longitudinal tire forces develop causing the vehicle to slow down. The vehicle slows down to about 5m/s from 20m/s. It can also be seen from the results that a rolling resistance moment is always present which causes the vehicle to naturally slow down when there is no braking torque applied to the wheels. The rolling resistance moment is a function of the normal force (refer to Equation (3.34)). When the vehicle is braking it pitches forward (vehicle is nose diving) as can be seen from the results, and thus load is being transferred to the front tires causing an increase in the rolling resistance moment being applied to the front tires. The pitch angle obtained from DynaFlexPro is the second component in the 321 Euler angle series used to track the orientation of the vehicle body with respect to the ground frame. Whereas a measure is created in ADAMS/View to calculate the pitch angle during simulation.

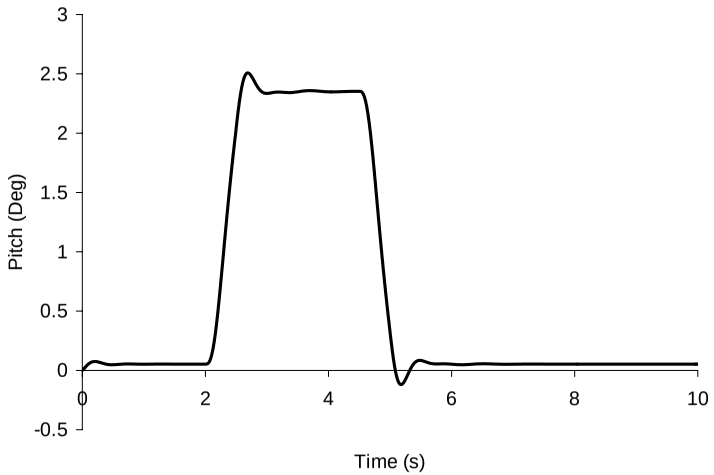
Excellent agreement is observed between DynaFlexPro and MSC.ADAMS as can be seen in the % difference curves shown in Figures (5.5) and (5.6). The % difference between the results obtained from both software packages is used to compare them and thus validate the results obtained from DynaFlexPro.



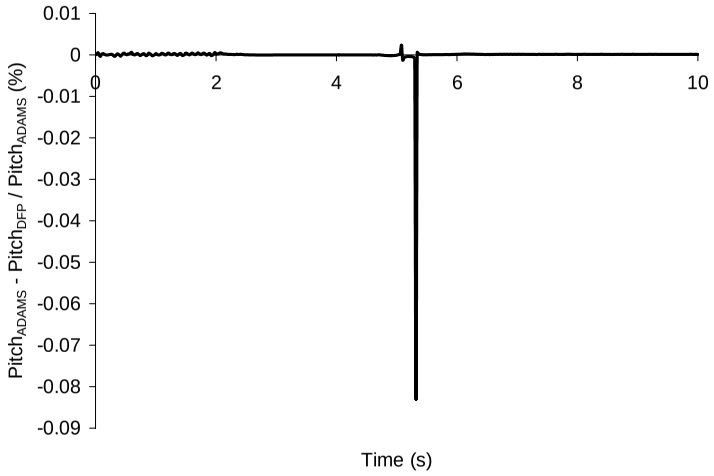
(a) Forward Velocity of the Vehicle Body



(b) Percent Difference in Calculating V_x

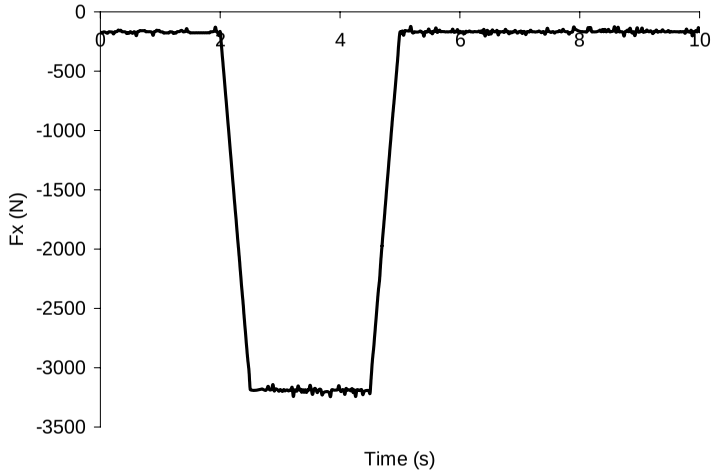


(c) Pitch Angle of the Vehicle Body

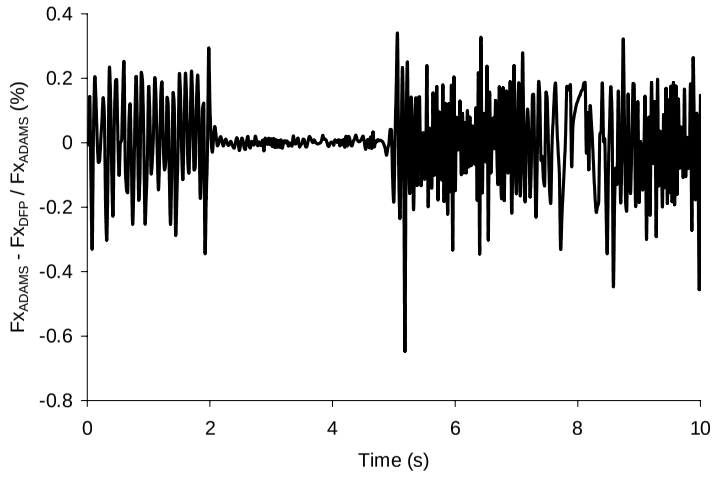


(d) Percent Difference in Calculating the Pitch Angle

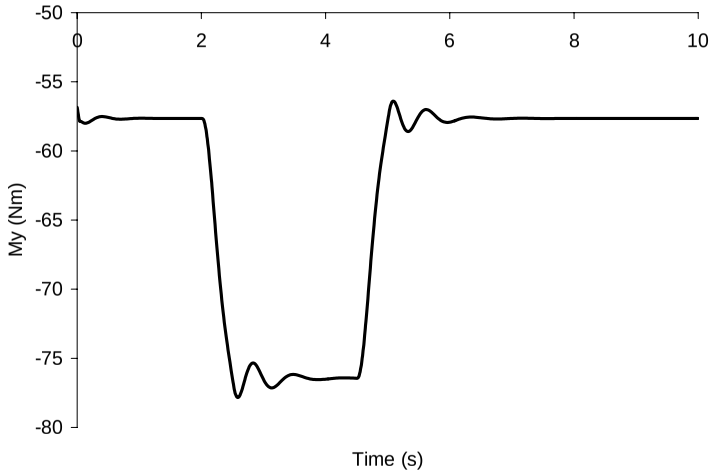
Figure 5.5: Response of the Chevrolet Equinox for the Example of a Braking Maneuver on a Flat Road



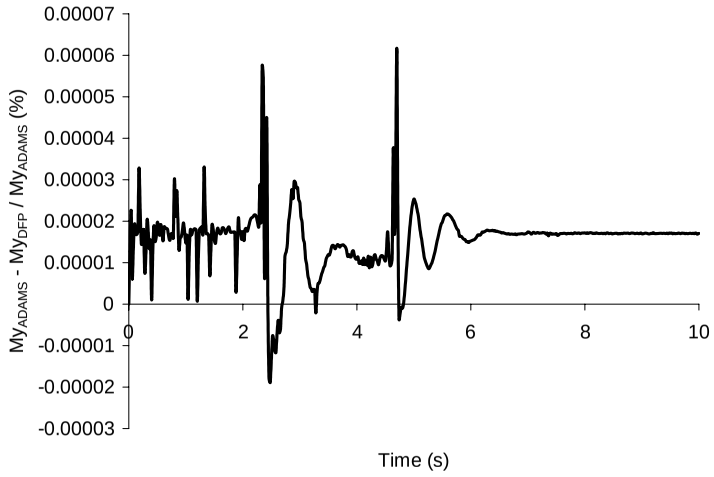
(a) Longitudinal Force of the Front Left (FL) Tire



(b) Percent Difference in Calculating F_x



(c) Rolling Resistance Moment of the FL Tire



(d) Percent Difference in Calculating M_y

Figure 5.6: Response of the Chevrolet Equinox for the Example of a Braking Maneuver on a Flat Road (Continued)

Note, in the % difference curves ADAMS indicates that the result is obtained from MSC.ADAMS and DFP indicates that the result is acquired from DFP. The % difference for each of the 4 results recorded is negligible. Similar results were obtained for other cases that are not recorded in this thesis. Thus, DynaFlexPro can be used with the tire/road component to analyze vehicle systems performing braking maneuvers on a flat road.

The simulation time is 12 times less for DynaFlexPro than for MSC.ADAMS. This implies that simulating the vehicle in DynaFlexPro can lead to a significant reduction in the time required to obtain the result. It can also be noted that the simulation time is a little bit faster when the tire/road component is used over the tire component. This can be possible since when the procedure is automatically generated around the $z = 0$ road plane, an optimized computational sequence is created that permits some of the tire intermediate variables to be calculated a bit faster than using the symbolic result (as is done in the tire component). However, the formulation time for tire/road component will be greater than that for the tire component since the road model procedure needs to be generated. This implies that the tire/road component can completely replace the tire component in DynaFlexPro and the $z = 0$ road plane used when the road is flat.

A convergence study was also performed by varying the tolerances of the numerical solver and by changing the numerical solver. It was found that the simulation results have converged when the tolerances were 0.001; the results at a tolerance of 0.0001 were essentially the same as the results at a tolerance of 0.001. Also, the rkf45 solver gave essentially the same results as the gear (based on Gear's method) solver. The criteria for the convergence study in both cases was the % difference between the results obtained between the cases with a different tolerance and with a another solver with a tolerance of 5%. It can also be noted that the simulation time is faster when the rkf45 solver is used over the gear solver and when the tolerance is 0.001 over 0.0001. Thus, the 4th-5th order Runge-Kutta numerical solver with a tolerance of 0.001 is a sufficient method to use to solve the system of ODEs for this example and is therefore used for all of the examples.

5.4.2 Double Lane Change Maneuver on a Hill

The second example involves the Chevrolet Equinox performing a double lane change maneuver as the vehicle goes over a hill. The double lane change maneuver is performed by applying an appropriate sine steer motion to the steering motion drivers. A single sine steer motion will perform a single lane change and thus a double sine steer motion is needed to perform a double lane change [101, 63, 52]. The period of each of the sine steer motions is 5s, where the amplitude of the first sine steer motion is 0.015 and that of the second is -0.01625. It can be noted that it is common that the second amplitude is higher than the first because the steering motion has to oppose the momentum of the vehicle [52]. There may also need to be a small additional steering input in between the two sine steer motions to adjust

the vehicle position so that it continues in a straight line when it is in the opposite lane before turning back to the desired lane [52]. Thus, a steering input of -0.012 is used to ensure that the vehicle travels in a straight line in the second lane. The steering input is shown in Figure (5.7) and in Equation (5.3).

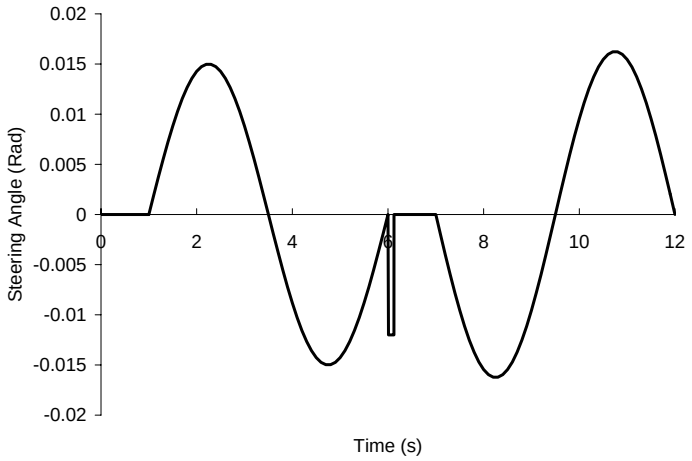


Figure 5.7: The Steering Motion Applied to Both Motion Drivers to Perform a Double Lane Change Maneuver

$$\text{steer(t)} = \begin{cases} & t & < 1s: & 0 \text{ rad} \\ 1s \leq & t & < 6s: & 0.015 \sin\left(\frac{2\pi(t-1)}{5}\right) \text{ rad} \\ 6s \leq & t & < 6.125s: & -0.012 \text{ rad} \\ 6.125s \leq & t & < 7s: & 0 \text{ rad} \\ & t & \geq 7s: & -0.01625 \sin\left(\frac{2\pi(t-7)}{5}\right) \text{ rad} \end{cases} \tag{5.3}$$

The steering motion is modeled in DynaFlexPro using Maple's piecewise function, and is modeled in MSC.ADAMS using the function builder in ADAMS/View by creating a series of if statements. This double lane change maneuver simulates the Equinox passing another vehicle.

The hill is modeled in DynaFlexPro as a sinusoidal function, and is shown in Equation (5.4) and in Figure (5.8).

$$z = f(x, y) = 5 \sin\left(\frac{2\pi(x + 1.487)}{400}\right) \tag{5.4}$$

The 3D spline road model in MSC.ADAMS allows a few known geometries to be superimposed on the specified centerline data for a higher degree of accuracy of the road profile [3]. One of these geometries is a hill modeled as a sinusoidal function. Thus, the hill is modeled in MSC.ADAMS using the 3D spline road model by superimposing the sinusoidal function representing the hill on top of a flat road. Therefore, the centerline of the road is specified in tabular form such that the

road remains parallel to the X axis with zero elevation, zero bank angle and with the left and right coefficients of friction being equal to 1, which is the same as in DynaFlexPro.

The 2002 Pacejka tire model is utilized to calculate the tire forces and moments with tire transients being considered. Thus the set of differential equations (Equations (3.23) and (3.24)) are appended to the 28 ODEs to give a set of 44 ODEs to represent the system. Furthermore, the relaxation lengths are calculated using the stretched sting equations suggested by Pacejka (Equations (3.26) and (3.27)). The list of parameters used in the 2002 Pacejka tire model and in the stretched string equations are presented in Appendix (B).

The example is simulated in both MSC.ADAMS and DynaFlexPro for a duration of 12s. The results from the example are shown in Figures (5.8)-(5.10) and in Table (5.2).

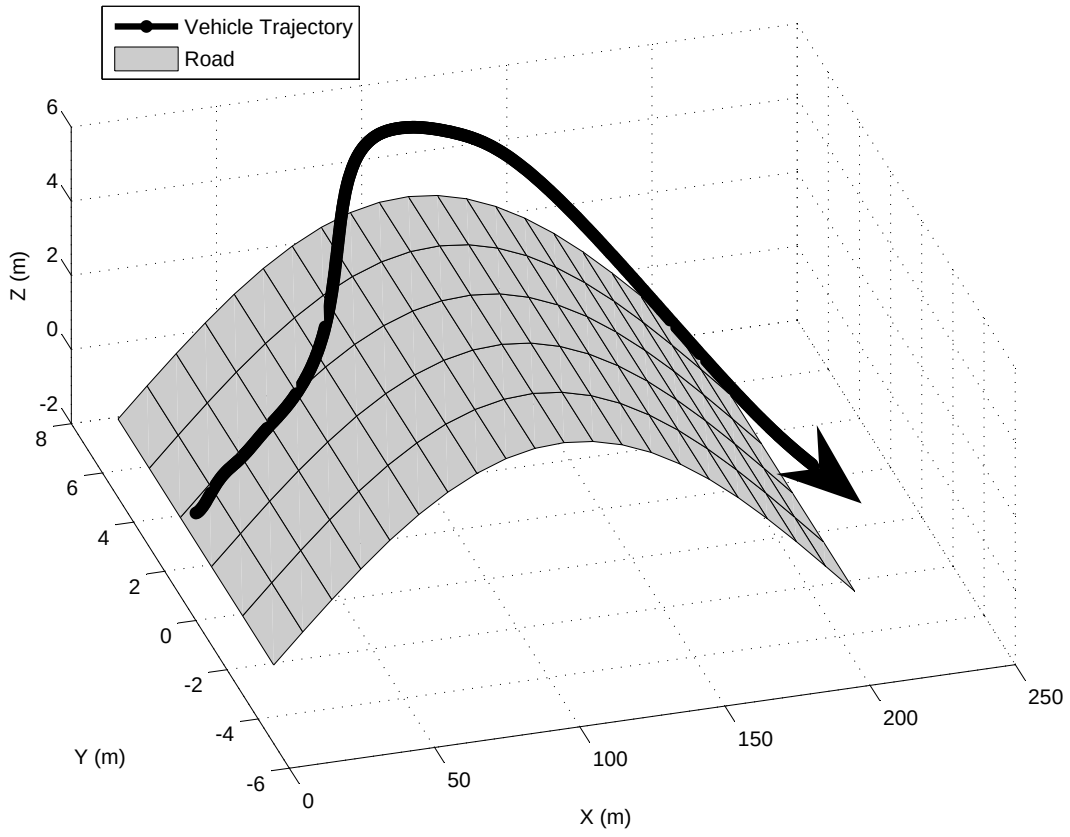
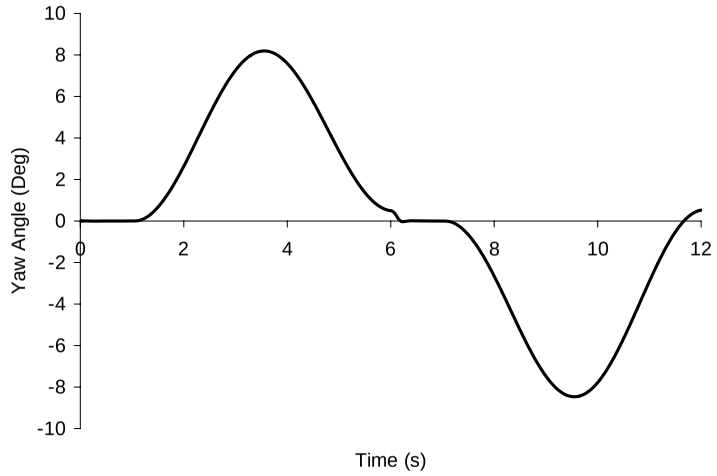
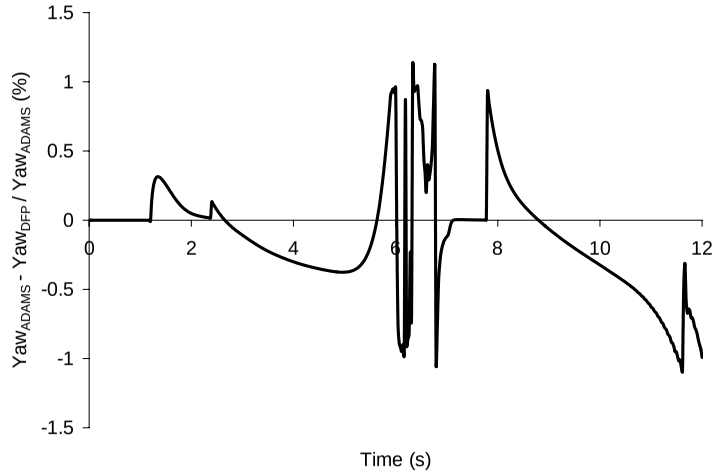


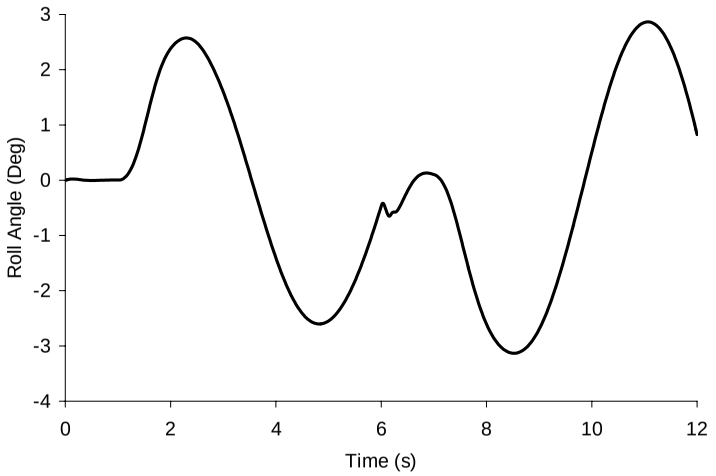
Figure 5.8: Vehicle Trajectory for the Example of the Chevrolet Equinox Performing a Double Lane Change Maneuver Over a Hill



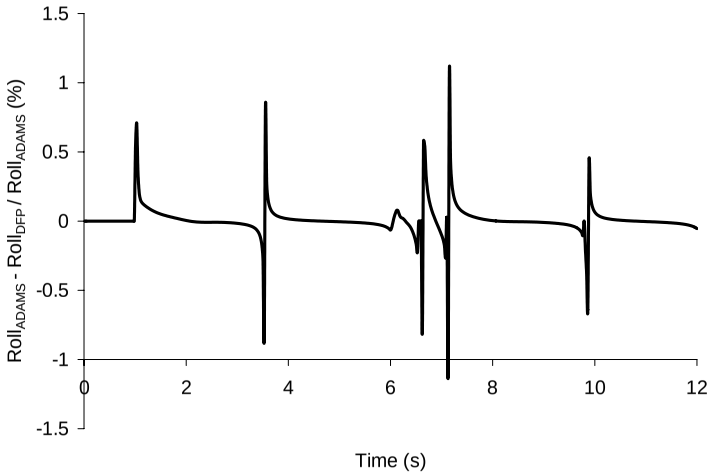
(a) Yaw Angle of the Vehicle Body



(b) Percent Difference in Calculating the Yaw Angle

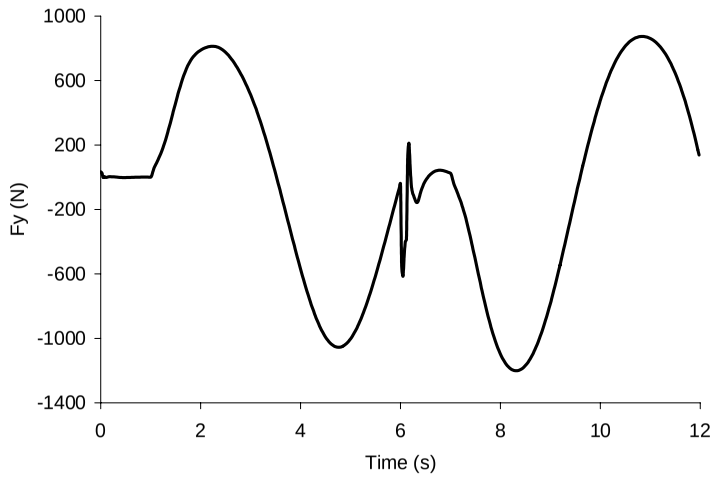


(c) Roll Angle of the Vehicle Body

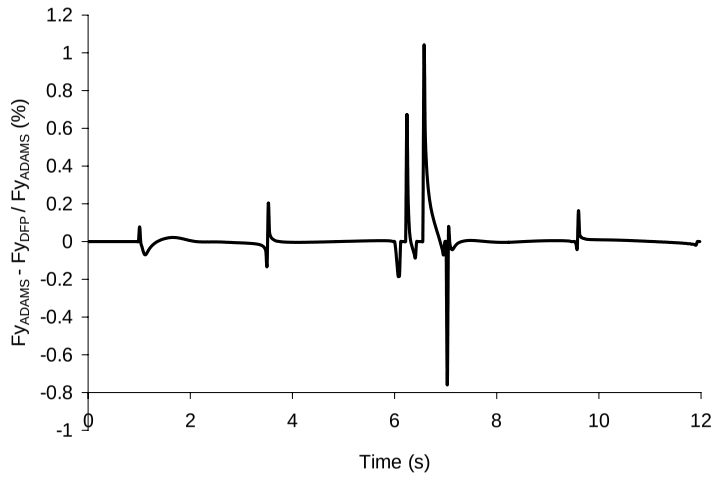


(d) Percent Difference in Calculating the Roll Angle

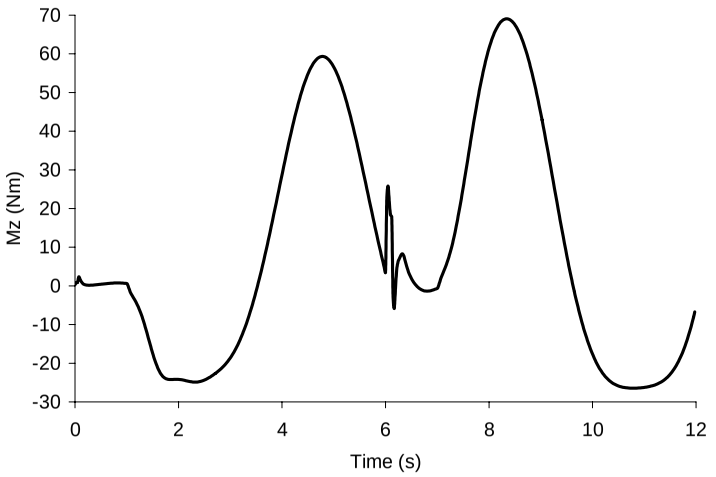
Figure 5.9: Response of the Chevrolet Equinox for the Example of a Double Lane Change Maneuver Over a Hill



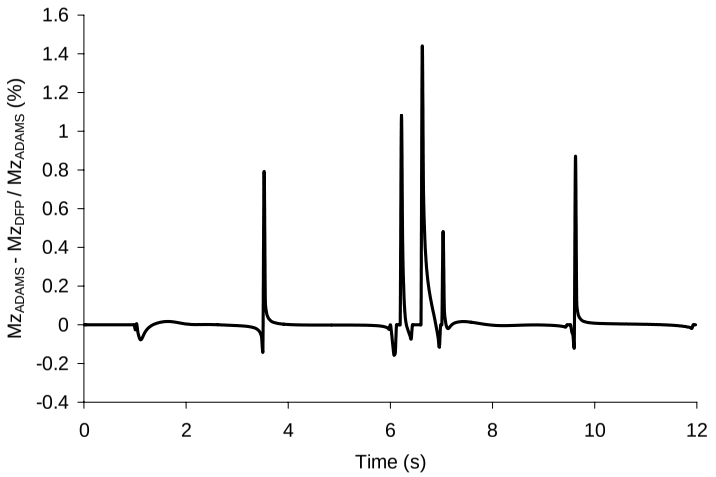
(a) Lateral Force of the Front Left (FL) Tire



(b) Percent Difference in Calculating F_y



(c) Aligning Moment of the FL Tire



(d) Percent Difference in Calculating M_z

Figure 5.10: Response of the Chevrolet Equinox for the Example of a Double Lane Change Maneuver Over a Hill (Continued)

Software	Simulation Time (s)
DynaFlexPro/Maple	5.12
MSC.ADAMS	45.99

Table 5.2: The Simulation Times for the Example of the Chevrolet Equinox Performing a Double Lane Change Maneuver Over a Hill

It can be seen from the results that the vehicle does indeed perform a double lane change as it travels over the hill. Note that the arrow in Figure (5.8) indicates the direction the vehicle is traveling in. The steering motion causes tire slip angles to develop, giving rise to lateral forces and aligning moments that cause the vehicle to yaw in order to perform the steering maneuver. The result for the yaw angle shows that the vehicle first turns to the second lane and then continues straight before turning back to the first lane. The yaw angle obtained from DynaFlexPro is the first component in the 321 Euler angle series, and it is obtained from MSC.ADAMS using a measure. The roll angle response is similar to the steering motion input as anticipated. The roll angle is obtained from MSC.ADAMS by creating a measure to calculate it during simulation, and it is the second component of the 321 Euler angle series that is used in DFP to track the orientation of the vehicle body.

When a vehicle corners it will roll towards the outside of the turn causing the normal force to increase on the outside tires and decrease on the inside ones [43]. The lateral force and the aligning moment are both dependent on the normal force. Thus, for the front left tire, the lateral force and the aligning moment should change according to the amount of vehicle roll and this can be seen in the results. The more the vehicle body rolls, the more the load should be transferred and the greater the lateral force and aligning moment should be. This can be seen in the results as the peak values of the vehicle roll are equivalent to the peak values of the lateral force and aligning moment.

Excellent agreement is observed between MSC.ADAMS and DFP as can be seen in the % difference curves shown in Figures (5.9) and (5.10). It can be noted that the trend of the % difference for the roll angle and for the lateral force and aligning moment are similar and this is because they are closely related. Thus, DynaFlexPro can be used with the tire/road component to obtain the response of vehicle systems while they are rolling over 3-D roads that can be modeled by a single elevation function and while they are performing steering maneuvers.

The simulation is 9 times faster in DynaFlexPro than in MSC.ADAMS. The simulation takes a bit longer (in relevance to the gain in time in comparison to MSC.ADAMS) to perform in DynaFlexPro than the previous example because of the added complexity of the road. However, a significant reduction in time is still achieved when the vehicle is simulated in DynaFlexPro using symbolic computing. It can also be noted that the simulation time is faster than the previous example (5.12s versus 13.5s) even with a greater simulation duration. The main reason for this is that the slip parameters are directly calculated by numerically integrating

Equations (3.26) and (3.27) based on a small set of intermediate parameters. Also, for this particular case the majority of the Pacejka tire model parameters are either 1 or 0 and thus simplify the calculation of the forces and moments.

5.4.3 Braking Maneuver on a Banked Road

The ADAMS/3D Road model is recommended when the road is complex and involves complicated geometries [90]. This road model permits arbitrary three-dimensional smooth road surfaces to be defined with complex geometries superimposed onto the road surface, such as parking structures, curbs, and ramps. MSC.ADAMS also has a road builder graphical user interface that can be used to construct the road [11]. The ADAMS/3D Road model also uses the 3D spline road model to determine the contact between the tire and the ground [11]. Furthermore, a driver model must be used with this road model to force the vehicle to follow the pre-defined trajectory [90].

However, the ADAMS/3D Road model can only be used when the vehicle is modeled in ADAMS/Car or ADAMS/Chassis [11]. The problem with ADAMS/Car and ADAMS/Chassis is that a full vehicle model is required to perform vehicle analysis on a 3D road trajectory. A full vehicle model is one that includes all the vehicle subsystem components (suspension (front and back), brakes, steering, transmission, motor, etc.). Thus, the simple vehicle model used to model the Chevrolet Equinox can not be used to model the vehicle in ADAMS/Car or ADAMS/Chassis.

Additionally, when the road is added to a model in ADAMS/View with the 3D spline road model, the capabilities of the 3D spline road model are not the same as they are when it is used in ADAMS/Car or ADAMS/Chassis. A driver model can not be used in ADAMS/View and with the 3D spline road model it is extremely important that the vehicle follows the path defined by the centerline of the road for the model to work correctly [3].

Therefore, it is difficult to model the vehicle rolling on a banked road surface in ADAMS/View. When the vehicle travels on a banked surface it will start to deviate from the centerline of the road because of the banked surface. This leads to problems in ADAMS/View since when determining the contact between the road and the tire, the 3D spline road model assumes that the centerline of the road is aligned with the center of the vehicle. This leads to a difference in the way that the bank angle is defined between each section of the road because the vehicle is deviating from the path; the bank angle should be defined with reference to the centerline of the road, but instead is being referenced somewhere else (centerline of the vehicle) since the vehicle is going off the path. This phenomena leads to false results and in some cases may cause MSC.ADAMS to stop because the integrator can not converge to the result.

The vehicle traveling on a banked road surface can be easily analyzed in DynaFlexPro using the tire/road component by specifying the banked surface as a single elevation function. The banked surface can also be tested in DynaFlexPro

by specifying the centerline of the road in tabular form and by defining the appropriate banked angle for each section of the road. Note that with this method a driver model should be used to force the vehicle to follow the centerline of the road. However, for this example the cubic spline coefficients will be 0 and the road will be defined solely using Equation (3.63). Thus, if the vehicle deviates from the path, the results obtained will still be valid. The braking performance of the Chevrolet Equinox is analyzed on a banked surface using both of the methods to define the road in order to validate the second approach to defining a banked road. For this example, let DFP1 indicate that the road is defined by a single elevation function and let DFP2 indicate that the road is defined in tabular form.

The road surface is defined as the $z = 0$ road plane banked about the global X axis by -3°. The single elevation function representing this road profile is shown in Equation (5.5).

$$z = f(x, y) = -0.0524y \tag{5.5}$$

A negative indicates that the elevation of the road is decreasing in the positive Y direction. The road profile is specified in tabular form by specifying the centerline of the road such that it always lies on top of the global X axis with zero elevation. The banked angle is also specified in the table representing the road as -0.0524rad for each section of the road. The second derivative of z with respect to s (the distance traveled along the path), zp_s, is specified as being equal to 0 at every data point.

The braking torque applied to each of the wheels to stop the vehicle is the same as was applied in the first example (Section (5.4.1)).

The example is performed in DynaFlexPro for both of the road profiles for a duration of 10s. The results from the simulation are shown in Figures (5.11)-(5.14) and in Table (5.3).

Software	Simulation Time (s)
DFP1/Maple	21.2
DFP2/Maple	31.0

Table 5.3: The Simulation Times for the Example of the Chevrolet Equinox Performing a Braking Maneuver on a Banked Road

The results indeed show that the vehicle does slow down on the banked road surface when the braking torque is applied to all four wheels. It can also be seen from the results that the vehicle does not stay in a straight line and starts to deviate from the centerline of the road even though the steering angle is kept at zero. This is because of the banked road. The banked road forces the vehicle to move in the direction of the banked road because the direction of the gravitational field is not perpendicular to the road surface. This leads to the generation of lateral forces that

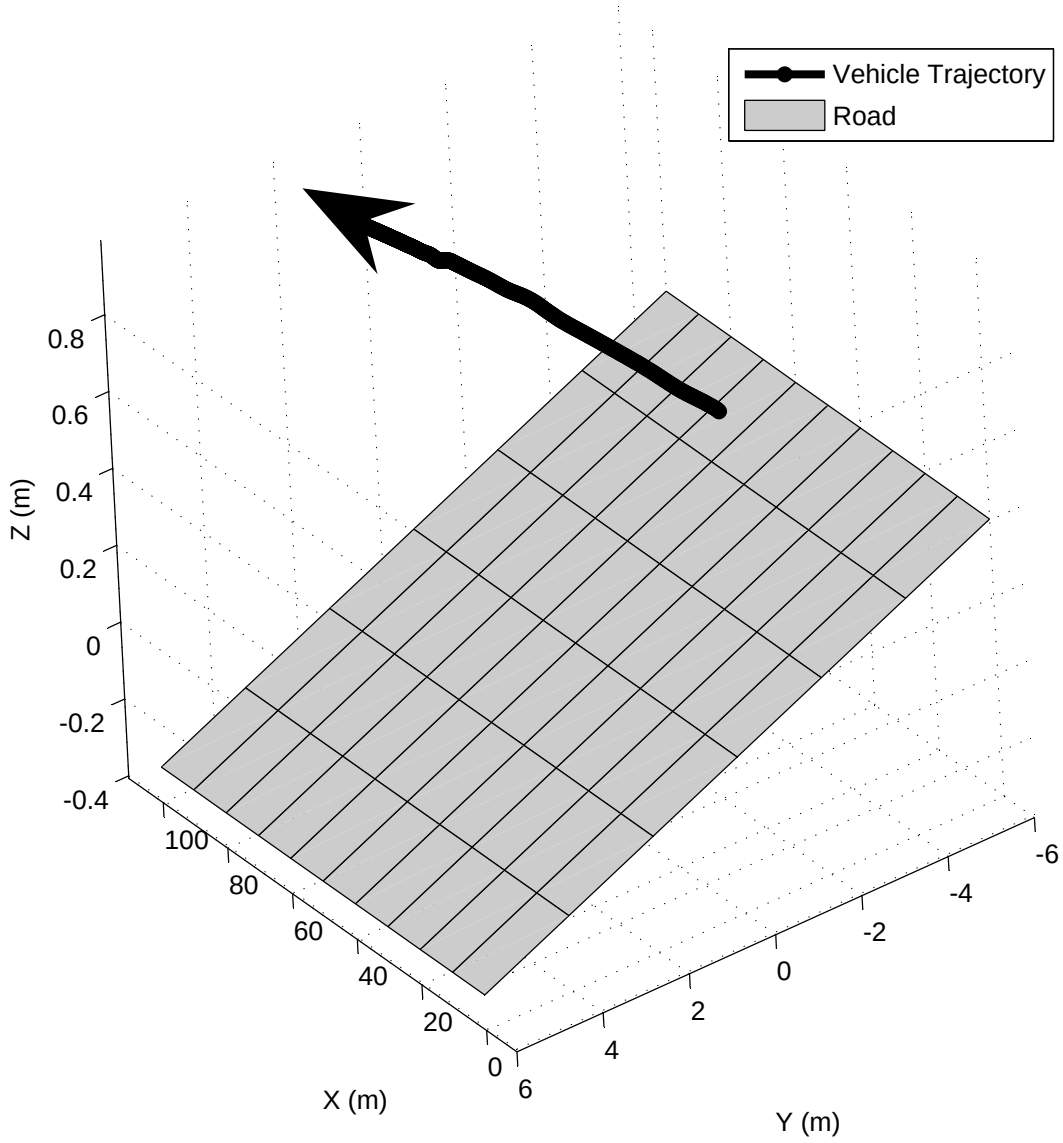
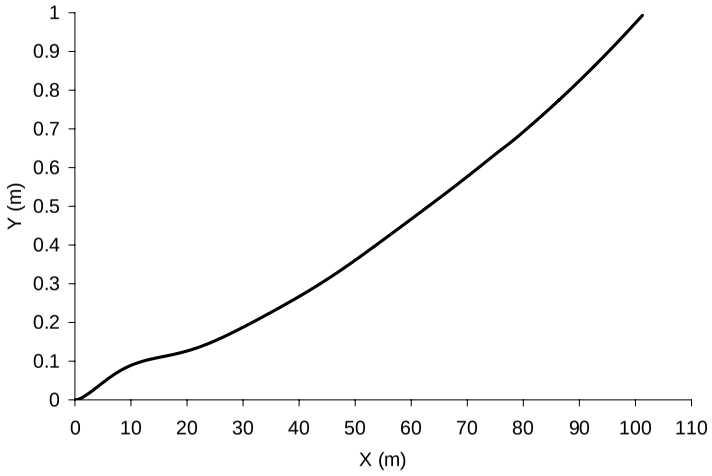
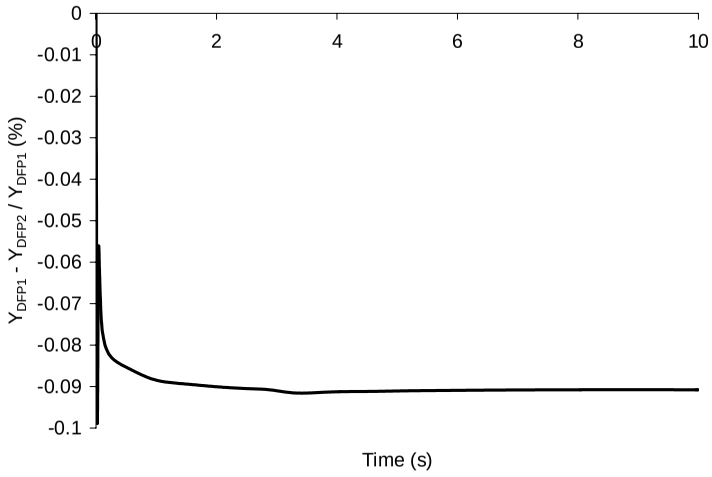


Figure 5.11: Vehicle Trajectory for the Example of the Chevrolet Equinox Performing a Braking Maneuver on a Banked Road

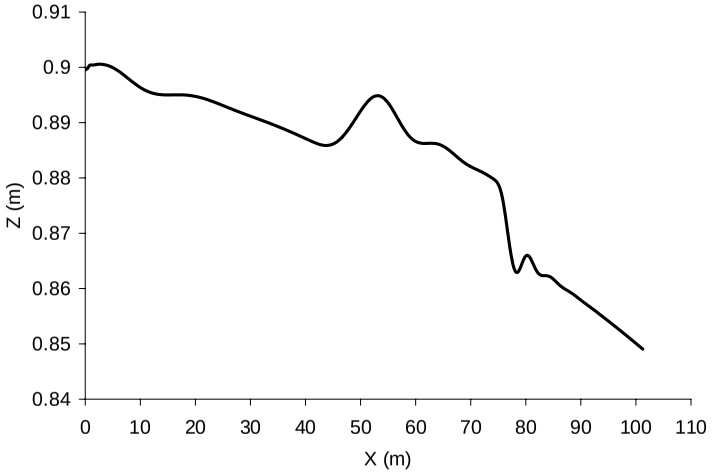
cause the vehicle to yaw (turn) in the direction of the banked road. The braking torque causes the vehicle to slow down to about 5m/s from 20m/s. Also, it can be noted that the response of the forward velocity of the vehicle is the same as that from the first example. This is because the bank angle of the road surface has no effect on the generation of the longitudinal forces and the forces generated are strictly dependent on the braking torque applied to the wheels. The vehicle also immediately starts to roll because of the presence of the bank angle and settles at a position of -3.9° (-0.068rad). The vehicle is rolling by a small amount (0.068rad - 0.0524rad = 0.056rad) because it is yawing in the direction of the bank angle.



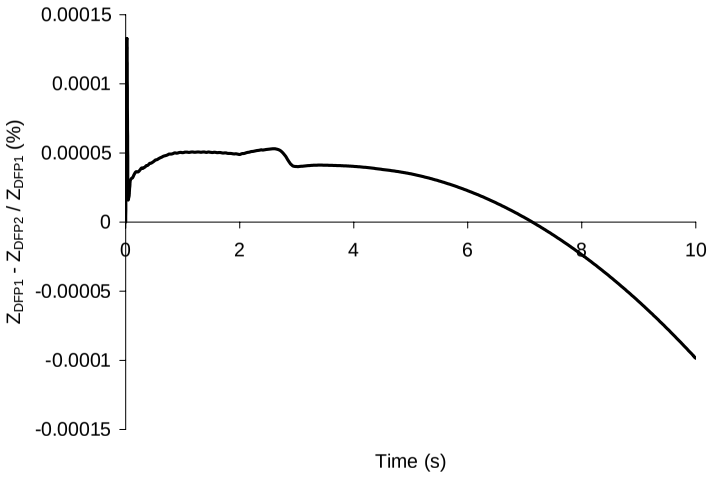
(a) Y Versus X Trajectory of the Vehicle Body



(b) Percent Difference in Calculating the Y Position of the Vehicle Body

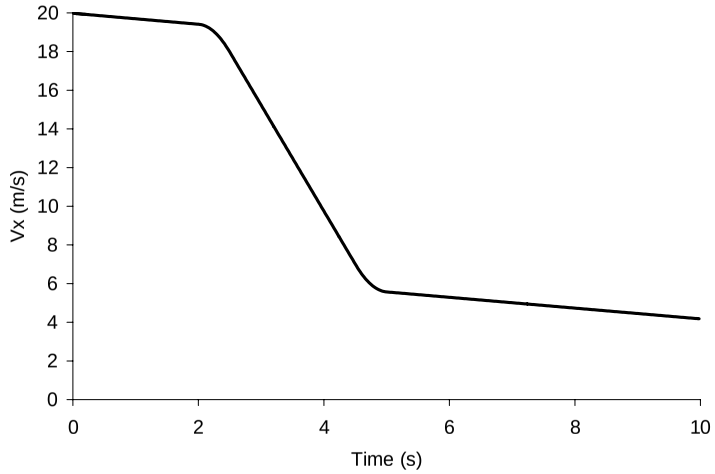


(c) Z Versus X Trajectory of the Vehicle Body

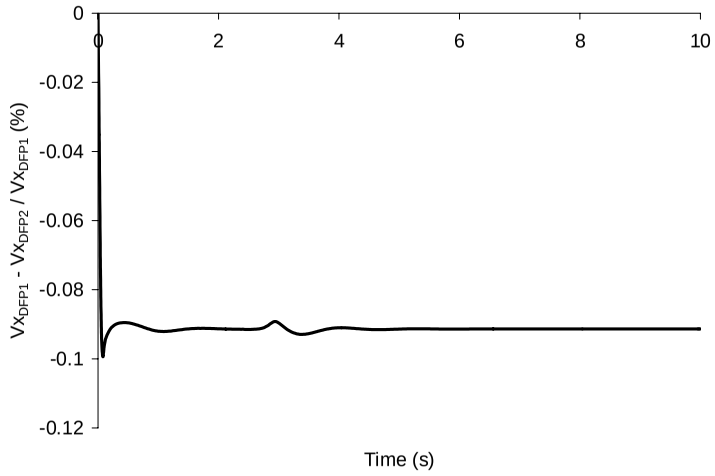


(d) Percent Difference in Calculating the Z Position of the Vehicle Body

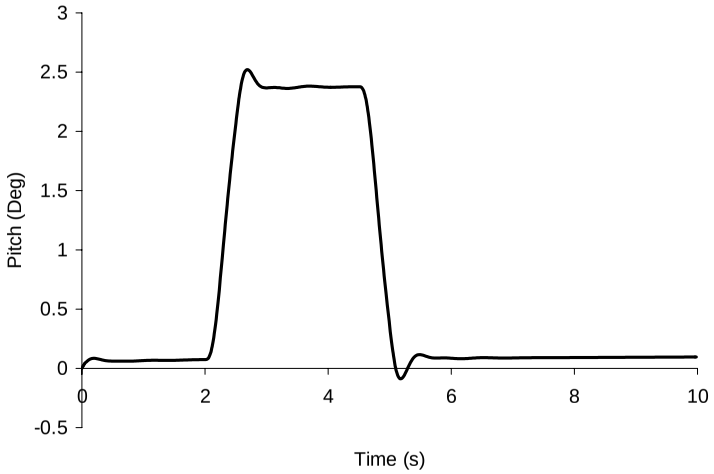
Figure 5.12: Response of the Chevrolet Equinox for the Example of a Braking Maneuver on a Banked Road



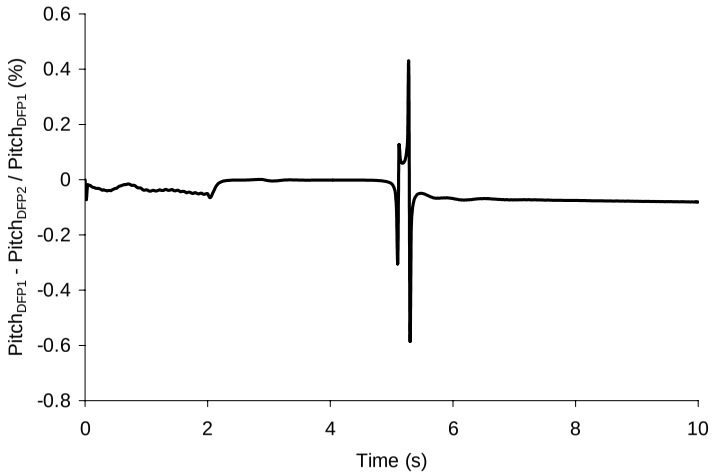
(a) Forward Velocity of the Vehicle Body



(b) Percent Difference in Calculating V_x

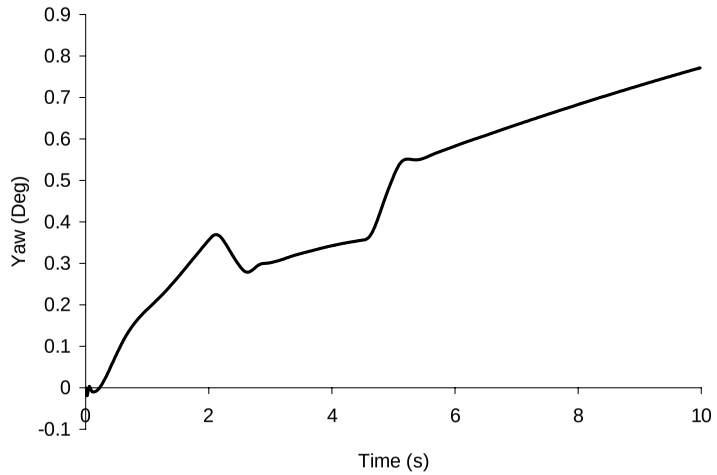


(c) Pitch Angle of the Vehicle Body

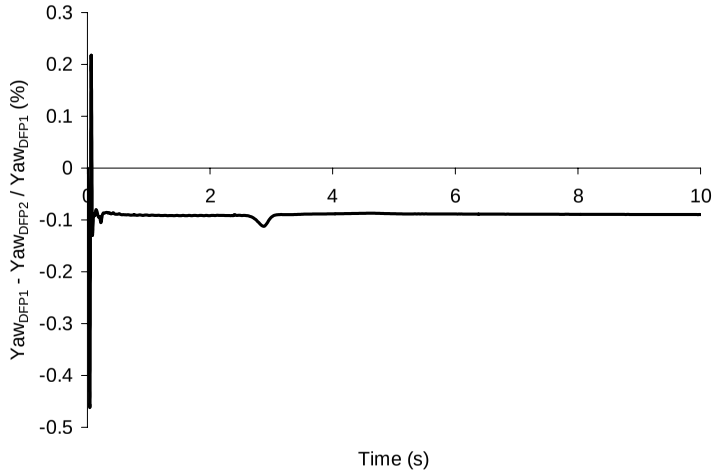


(d) Percent Difference in Calculating the Pitch Angle

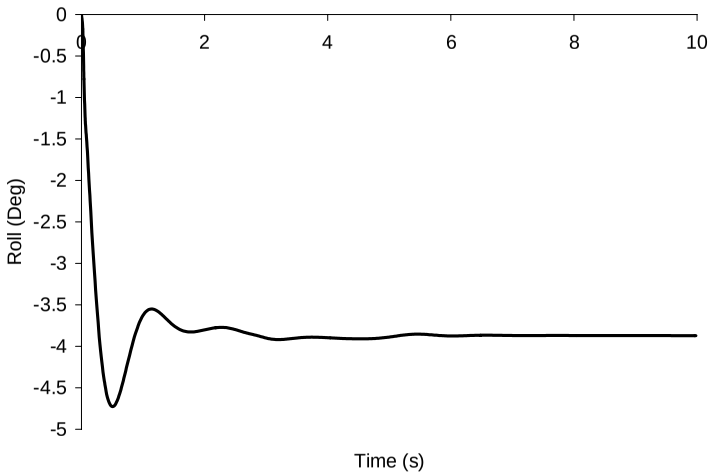
Figure 5.13: Response of the Chevrolet Equinox for the Example of a Braking Maneuver on a Banked Road (Continued)



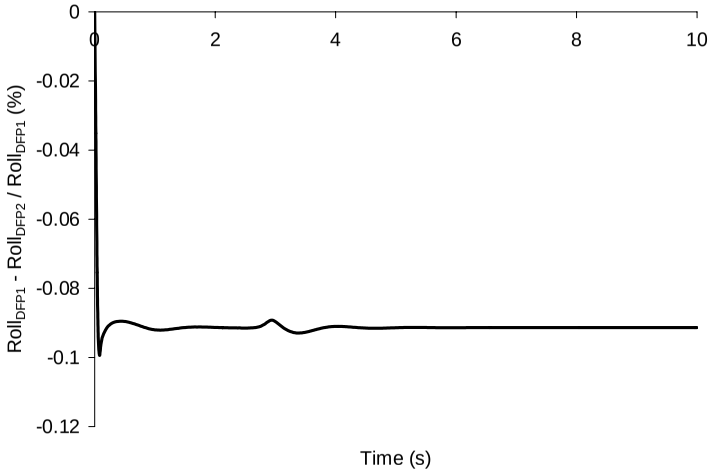
(a) Yaw Angle of the Vehicle Body



(b) Percent Difference in Calculating the Yaw Angle



(c) Roll Angle of the Vehicle Body



(d) Percent Difference in Calculating the Roll Angle

Figure 5.14: Response of the Chevrolet Equinox for the Example of a Braking Maneuver on a Banked Road (Continued)

The % difference curves shown in Figures (5.12)-(5.14) show that the results obtained from DynaFlexPro for both methods to represent the road are in excellent agreement. Thus, this example confirms that DynaFlexPro with the tire/road component can be used to analyze vehicles on banked road surface by either representing the road surface as a single elevation function or by representing it in tabular form. Furthermore, this example shows that DynaFlexPro with the tire/road component can be used to analyze the vehicle no matter the fidelity of the vehicle model.

The simulation is 1.5 times faster in DynaFlexPro when the road is modeled as an elevation function than when it is modeled in tabular form. The main reason why the model is slower when the road is defined in tabular form is that an additional function call is required at each time step for each tire to locate the current position of the tire inside of the table. It can also be noted that the simulation is about 1.5 times slower than the first example for the case when the road is defined as a single elevation function. The only difference between the two examples is the road profile. Thus, the simulation time is dependent on the complexity of the elevation function used to define the road and on the method utilized to represent the road.

5.5 Roads Not Represented by a Single Elevation Function

The second set of examples presented are ones where the road can not be modeled as a single elevation function and thus needs to be defined in tabular form. Three examples are considered to justify the tire/road component when the road is defined in tabular form and they are as follows: the vehicle executes a single lane change maneuver as it climbs an inclined surface after traveling on a flat surface, the vehicle performs a constant radius cornering maneuver, and the vehicle travels along a pre-defined track. For each of the cases, the road model is pre-defined based on the piecewise cubic spline function used to represent the road profile in each section of the road, and the spline coefficients are calculated during the formulation stage.

5.5.1 Lane Change Maneuver on an Inclined Surface

The fourth example involves the vehicle performing a single lane change maneuver as it is climbing a hill of 10% grade (5.7°). The single lane change maneuver is performed by applying a sine steer motion to the motion drivers with an amplitude of 0.015 and a period of 5s. The steering motion applied to both motion drivers is shown in Figure (5.15) and Equation (5.6).

$$\text{steer(t)} = \begin{cases} t < 5s: & 0 \text{ rad} \\ 5s \leq t < 10s: & 0.015 \sin\left(\frac{2\pi(t)}{5}\right) \text{ rad} \\ 10s \leq t < 10.3s: & -0.02 \text{ rad} \\ t \geq 10.3s: & 0 \text{ rad} \end{cases} \tag{5.6}$$

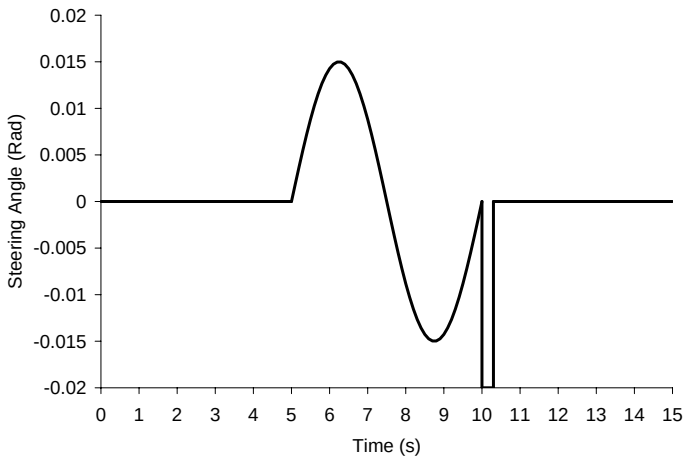


Figure 5.15: The Steering Motion Applied to Both Motion Drivers to Perform a Single Lane Change Maneuver

Note, the steering motion is initially zero because the vehicle does not perform the steering maneuver until it starts to climb the hill. When the vehicle is done performing the steering motion the steering returns to zero. However, before the steering returns to zero, a small negative steering angle is used to correct the position of the vehicle so that it continues in a straight line in the second lane. A similar steering motion was used to correct the vehicle in the second example (Section (5.4.2)). The steering motion is modeled in DynaFlexPro using Maple's piecewise function, and is modeled in MSC.ADAMS using the function builder in ADAMS/View by creating a series of if statements. This single lane change maneuver can simulate the vehicle having to change a lane due to construction or due to the fact that one of the road lanes is merging into another.

The first portion of the road surface is flat and the second is a hill with a 10% grade. Thus, the road can be represented as $z = 0$ for the first portion and as $z = 0.1(x - 80)$ for the second portion as can be seen in Equation (5.7).

$$z = f(x, y) = \begin{cases} x < 80\text{m} & 0 \text{ m} \\ x \geq 80\text{m} & 0.1(x - 80) \text{ m} \end{cases} \tag{5.7}$$

Note, it is at $x = 80$m that the road changes. However, the complete road surface can not be represented as a single elevation function and thus needs to be defined in tabular form. It can be noted that the road profile is independent of y. Thus, a piecewise cubic spline in terms of x can be used to represent the road in each section (x replaces s in Equation (3.66)). The road is defined in tabular form by specifying x, z, zp_x, and α at each data point. Note, zp_x is the second derivative of z with respect to x and α is the bank angle. The table used to define the road for this example is shown in Appendix (D). The road is specified in MSC.ADAMS in a similar fashion as it was specified for the second example (Section (5.4.2)). The only difference is that instead of superimposing a sinusoidal function onto the road,

a ramp is superimposed onto the road at $x = 80$m. The complete road data file used to represent the road in MSC.ADAMS is shown in Appendix (D).

The example is simulated in both MSC.ADAMS and DynaFlexPro for a duration of 15s. The results from the example are shown in Figures (5.16)-(5.18) and in Table (5.4).

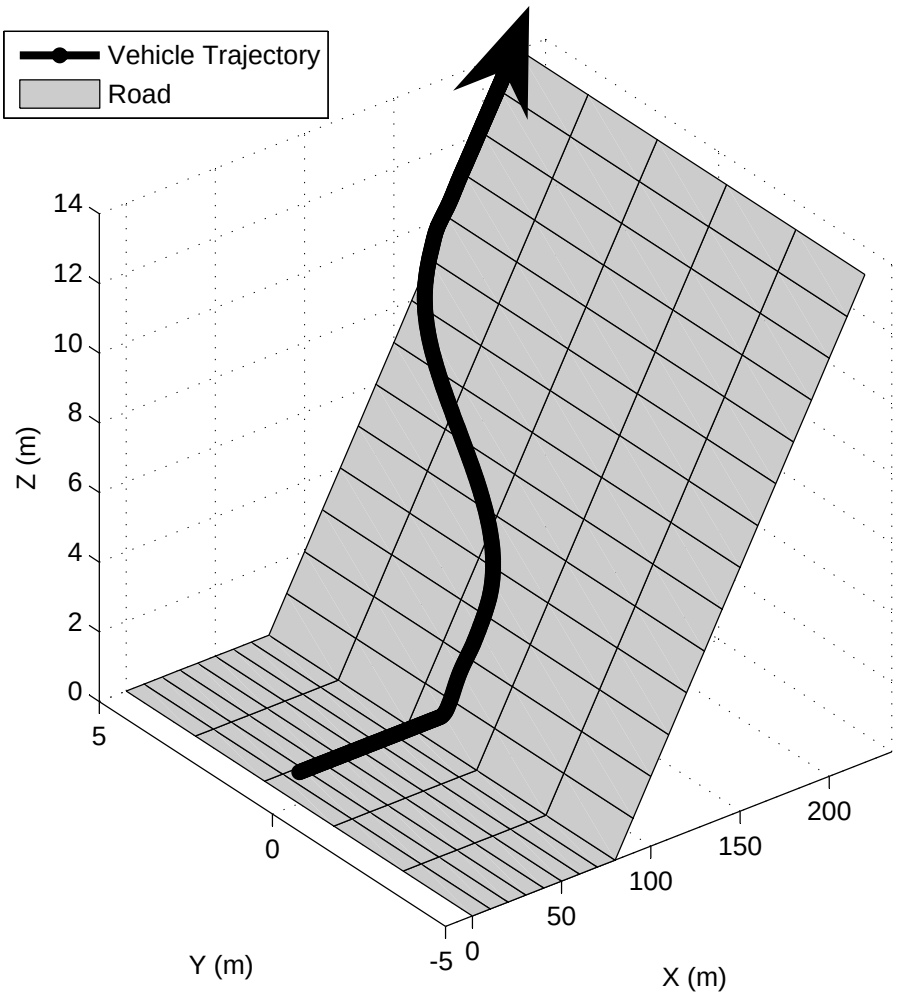
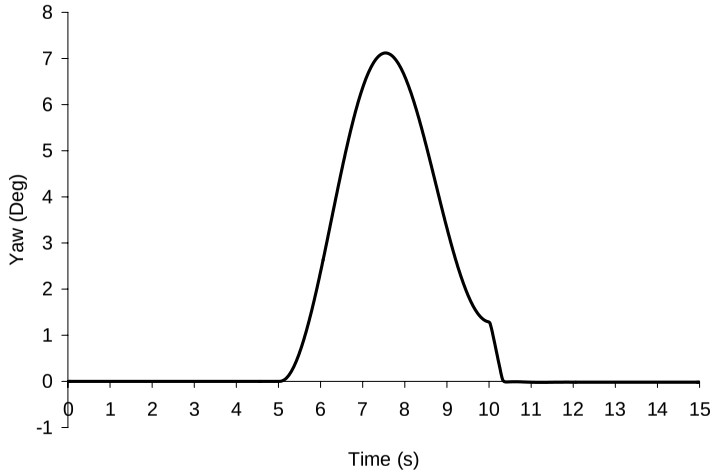


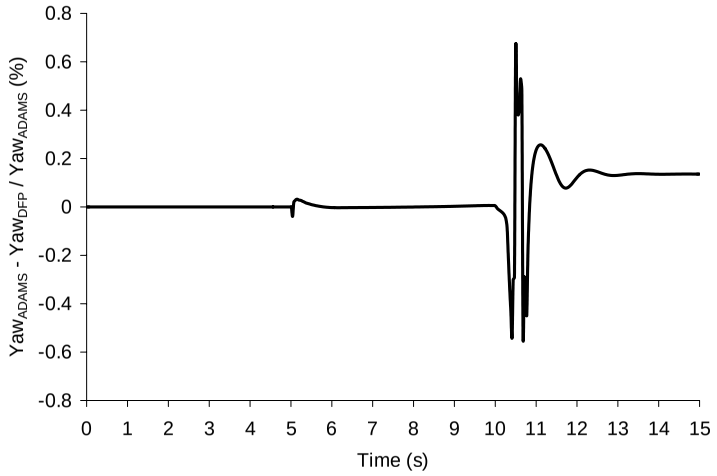
Figure 5.16: Vehicle Trajectory for the Example of the Chevrolet Equinox Performing a Single Lane Change on a Hill with a 10% Grade

Software	Simulation Time (s)
DynaFlexPro/Maple	18.1
MSC.ADAMS	107.6

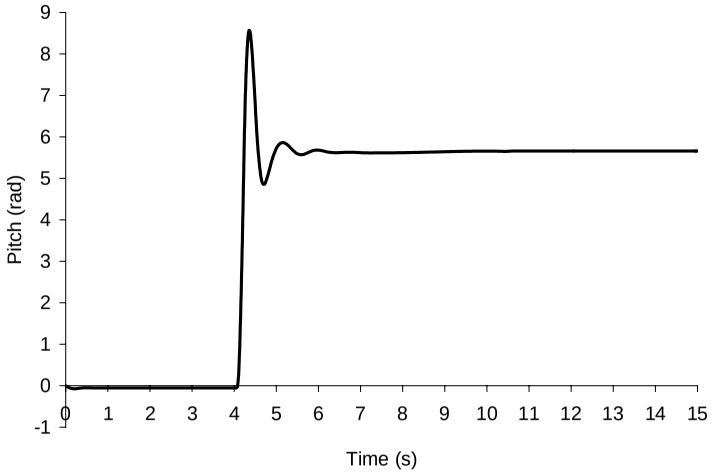
Table 5.4: The Simulation Times for the Example of the Chevrolet Equinox Performing a Single Lane Change on a Hill with a 10% Grade



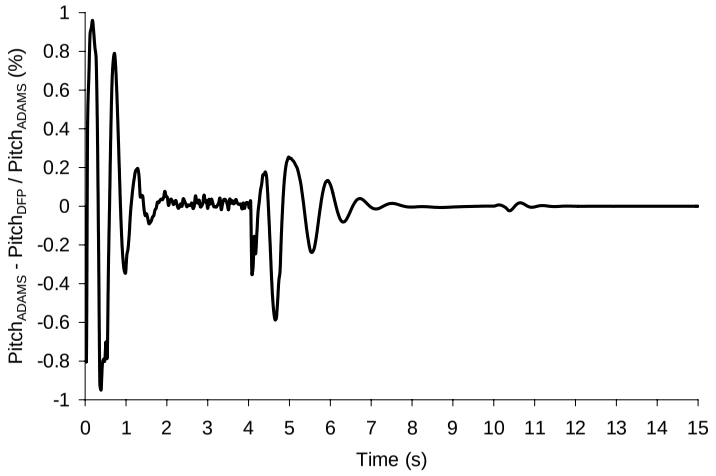
(a) Yaw Angle of the Vehicle Body



(b) Percent Difference in Calculating the Yaw Angle

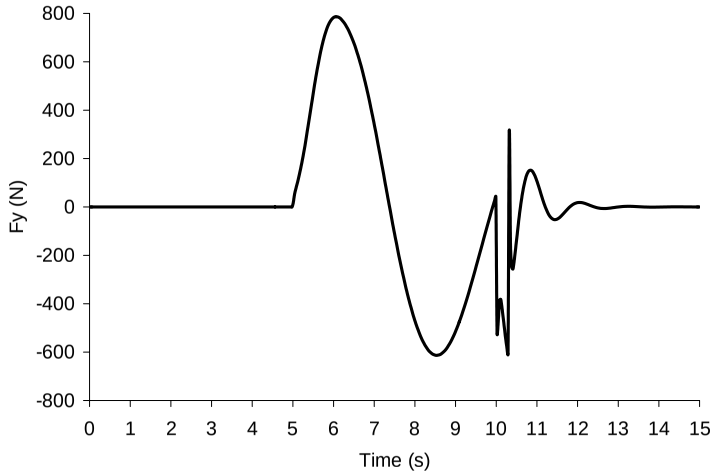


(c) Pitch Angle of the Vehicle Body

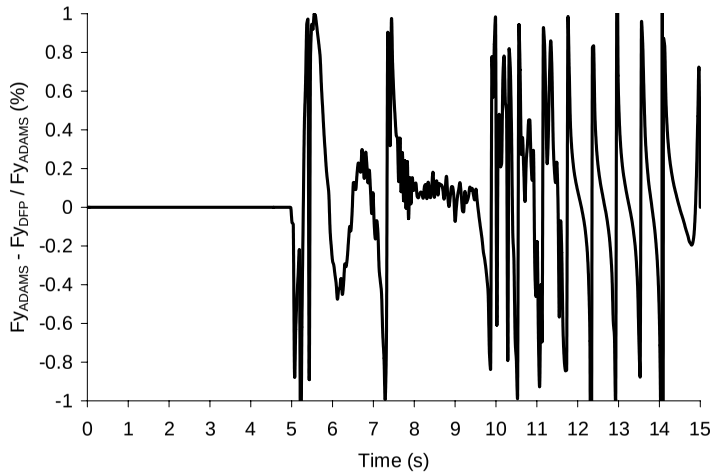


(d) Percent Difference in Calculating the Pitch Angle

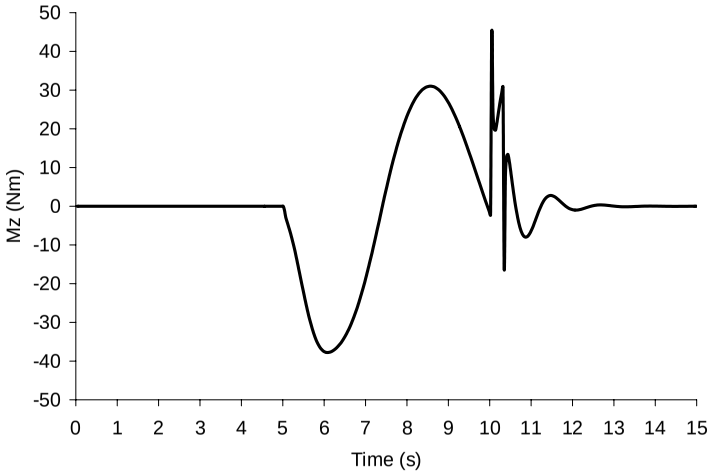
Figure 5.17: Response of the Chevrolet Equinox for the Example of a Single Lane Change Maneuver on a Hill with a 10% Grade



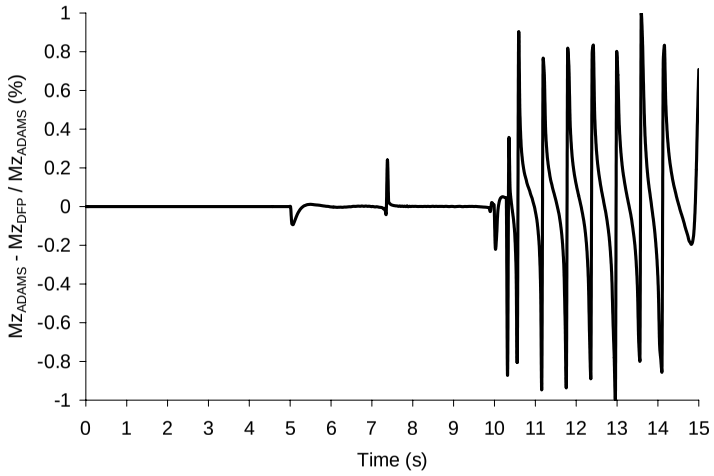
(a) Lateral Force of the Front Left (FL) Tire



(b) Percent Difference in Calculating F_y



(c) Aligning Moment of the FL Tire



(d) Percent Difference in Calculating M_z

Figure 5.18: Response of the Chevrolet Equinox for the Example of a Single Lane Change Maneuver on a Hill with a 10% Grade (Continued)

The vehicle does indeed perform a lane change maneuver shortly after it starts to climb the inclined surface. It can be noted from the response of the yaw angle that the additional steering angle is required to correct the vehicle so that it continues in a straight line in the second lane. The results for the vehicle trajectory confirm that the vehicle does continue in a straight line after it performs the lane change. The steering motion causes the tires to slip in the lateral direction which leads to the development of lateral forces and aligning moments that cause the vehicle to yaw. The steering motion can be seen in the response of the lateral force and aligning moment of the front left tire confirming their role in yawing the vehicle so that it performs the necessary steering maneuver. The vehicle pitch angle shows that the vehicle does start to climb the inclined surface when it encounters it. It can also be noted that the pitch angle settles to a value of 5.7° which is expected because this is the angle that the surface is inclined to the flat road. It is also important to note that the forward velocity of the vehicle will start to decrease at a faster pace when the vehicle is on the inclined surface than on the flat surface because of the presence of the gravitational field. The rolling resistance moment acts to slow down the vehicle when it is rolling on the flat surface and the rolling resistance moment and the gravitational field act to slow down the vehicle when it is rolling upwards on the inclined surface.

The % difference curves show that there is an excellent agreement between the results obtained from DynaFlexPro and from MSC.ADAMS. Therefore, DynaFlexPro with the tire/road component can be used to analyze wheeled vehicle systems on a road that only varies with respect to the x direction by representing the road in tabular form.

The example also shows that the vehicle can be simulated in DFP 5.9 times faster than in MSC.ADAMS. The simulation time (in relevance to the gain in comparison to MSC.ADAMS) is slower than the other examples because the road needs to be defined in tabular form. Defining the road in tabular form introduces another numerical iteration for each tire at each time step in order to locate the current position of each tire within the table representing the road. However, a significant reduction is still obtained when the vehicle is simulated in a symbolic environment.

5.5.2 Cornering Maneuver on a Pre-Determined Track

This example involves the vehicle performing a constant radius cornering maneuver at constant speed. This maneuver is typically used in vehicle dynamic studies to determine the handling characteristics of the vehicle in a turn [68]. The steering motion applied to both of the motion drivers for this example is a motion that forces the vehicle to remain on the centerline of the road that is defined by a circle, which is necessary because a driver model is not being used. The centerline of the road is defined as a circle in the XY plane with a radius of 100m that is centered at X=0m and Y=100m. The steering angle required to keep the vehicle on the

centerline of the path is a constant 0.0365 radians. Thus, this steering motion is applied to both of the motion drivers.

There needs to be a torque applied to the wheels to keep the forward velocity of the vehicle constant because of the presence of the rolling resistance moment. A proportional controller can be used to adjust the torque applied to the wheels so that the velocity of the vehicle remains constant. Essentially, a torque is applied to the wheels when there is a difference between the desired velocity of the vehicle and the actual velocity. A proportional constant is used to convert the velocity difference to a torque. The torque applied to all of the wheels is shown in Equation (5.8).

$$\text{torque(t)} = kp(V_{x\_\text{desired}} - V_{x\_\text{actual}}) = 2000(20 - V_x(t)) \tag{5.8}$$

Note, the proportional constant is kp.

The constant radius cornering maneuver is a maneuver that is performed on a flat surface. Thus, the surface can be modeled as a single elevation function as was done in the first example (Section (5.4.1)). However, the centerline of the road will be specified in tabular form to justify the approach by comparing the results with MSC.ADAMS. Since the road is varying in both the x and y directions, the distance traveled along the path must be the independent variable for the piecewise cubic spline that is used to represent each section of the road. The road is thus defined in tabular form by specifying x, y, z, zp_s, and α for each data point. The table that is used to define the road is shown in Appendix (D). Note, for this example the vehicle is only going to travel along the circle for a distance that is equal to a quarter of the circumference. Thus, only this section of the road is specified in the table.

The 3D spline road model can not be used to define the road in MSC.ADAMS because of reasons discussed in Section (5.4.3). The main reason for this is that a driver model can not be used in ADAMS/View. As previously specified it is extremely important that the vehicle follow the path defined by the centerline of the road. Initially, the vehicle will be slightly deviated from the path because of the sudden yaw rate being applied to the vehicle when it starts the constant radius steering maneuver. Thus, the 2D road model is used with the road type specified as flat as was done in the first example (Section (5.4.1)).

The example is performed in MSC.ADAMS and DynaFlexPro for a simulation duration of 7.85s. Note, the vehicle is only going to travel around a quarter of the circumference of the circle that is used to define the centerline of the road and the velocity of the vehicle is maintained at a constant; thus 7.85s is sufficient to perform this maneuver as is shown in Equation (5.9).

$$t = \frac{C/4}{V_x} = \frac{\frac{\pi D}{4}}{V_x} = \frac{\frac{\pi 200m}{4}}{20m/s} = 7.85s \tag{5.9}$$

The results from the example are shown in Figures (5.19)-(5.22) and in Table (5.5).

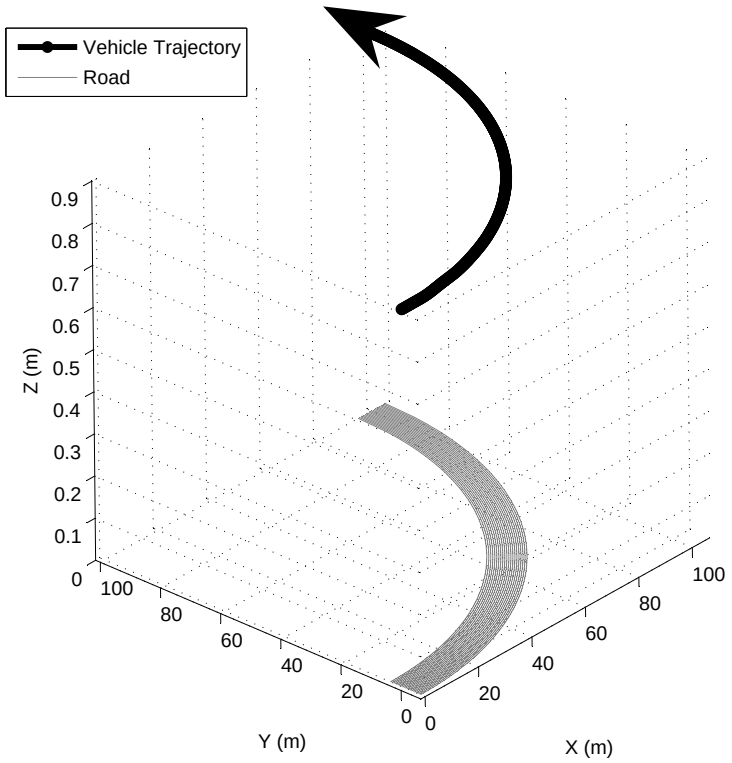


Figure 5.19: Vehicle Trajectory for the Example of the Chevrolet Equinox Performing a Constant Radius Cornering Maneuver

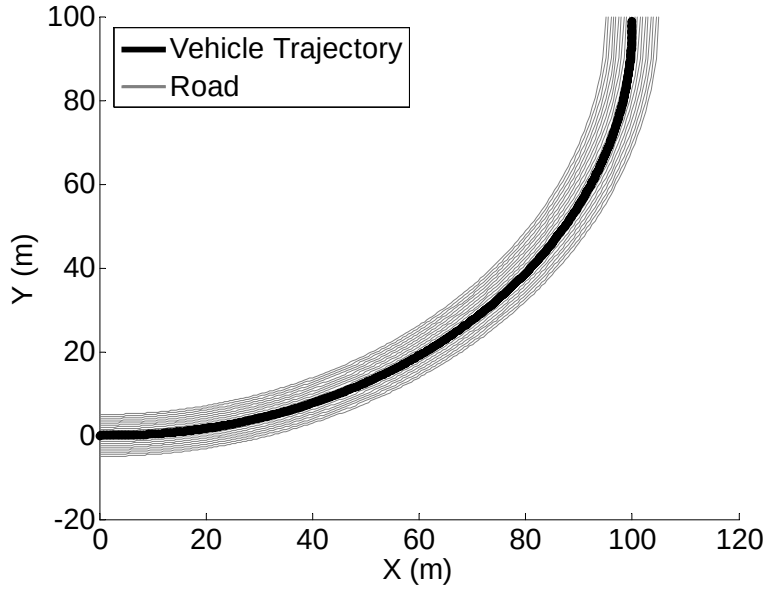


Figure 5.20: Top View of Vehicle Trajectory for the Example of the Chevrolet Equinox Performing a Constant Radius Cornering Maneuver

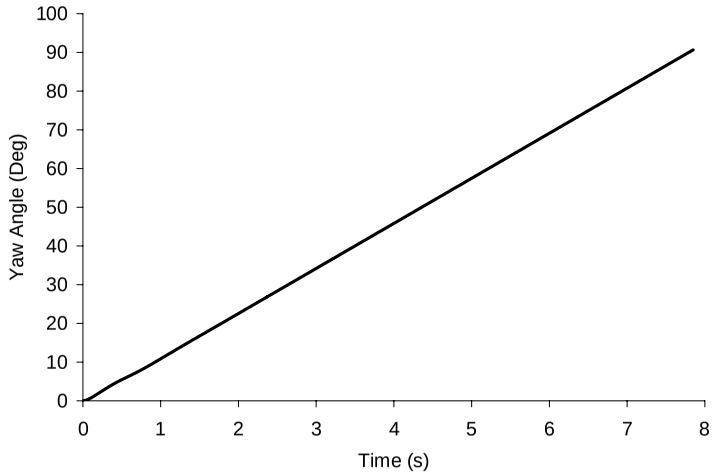
Software	Simulation Time (s)
DynaFlexPro/Maple	24.5
MSC.ADAMS	90.6

Table 5.5: The Simulation Times for the Example of the Chevrolet Equinox Performing a Constant Radius Cornering Maneuver

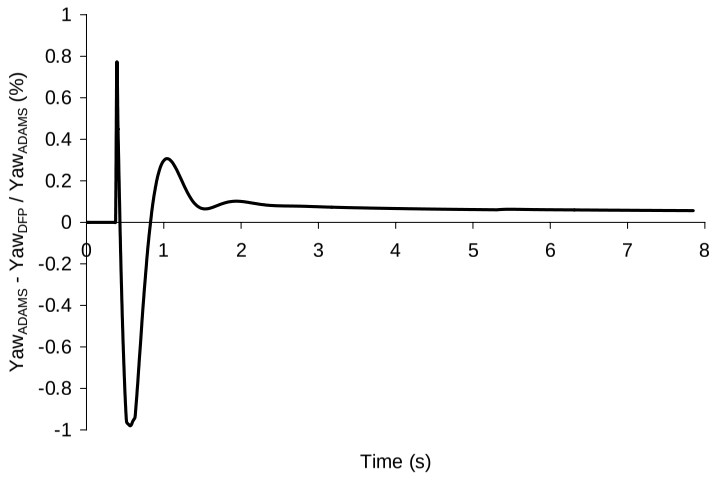
The results show that the vehicle does perform a constant radius cornering maneuver around the road defined by a quarter of a circle in the XY plane with a radius of 100m. This can be seen in the response of the vehicle trajectory. The response of the top view of the vehicle trajectory shows that the vehicle does follow the centerline of the road. The yaw angle response assures that the vehicle has traveled a quarter distance around the circle as the final yaw angle is 90°. The yaw angle response also confirms that the the proportional controller is sufficient to keep the velocity of the vehicle constant, since the vehicle completed the maneuver in 7.85s. The results also show that the constant steering angle gives rise to a constant lateral force and aligning moment that cause the vehicle to yaw in order to perform the cornering maneuver. It can further be seen that the roll angle of the body is constant which causes the tire normal forces to remain constant (vehicle roll angle is not changing so forces are not being transferred between the left and the right side of the vehicle). Also, the forward vehicle velocity is held constant, thus the tire slip angles will also remain constant. Therefore, it does indeed make sense that the lateral force and aligning moment remain constant throughout the steady state cornering maneuver.

Excellent agreement is achieved between the results obtained from MSC.ADAMS and from DynaFlexPro as can be seen in the % difference curves shown in Figures (5.21) and (5.22). Thus, DynaFlexPro with the tire/road component can be used to analyze vehicle systems on a 3-D road that is defined by specifying the centerline of the road in tabular form.

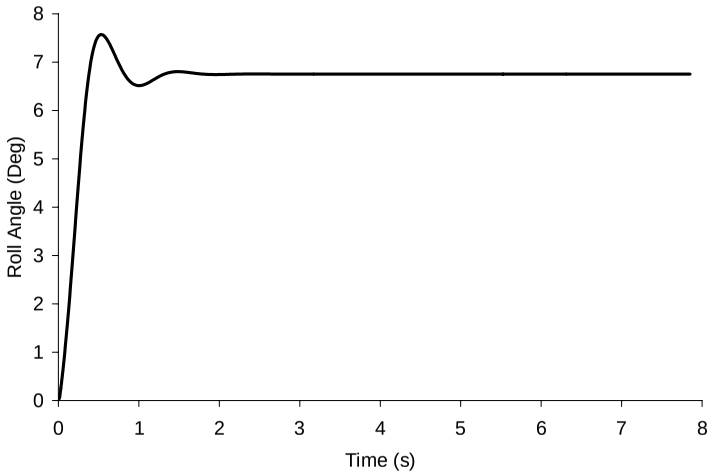
The simulation is 3.7 times faster when simulated in DFP than when simulated in MSC.ADAMS. It can be noted that this example shows the lowest gain in simulation time when comparing the times in DFP and MSC.ADAMS. This is because the road model used in MSC.ADAMS is a much simpler road model than the one used in DFP. For this example, the road could have been modeled as a single elevation function ($z = 0$) instead of being represented in tabular form. If the road would have been represented as a single elevation function then a much larger gain would have been achieved as in the first example (Section (5.4.1)). Thus, to achieve a good simulation time, the road should be represented as a single elevation function when possible.



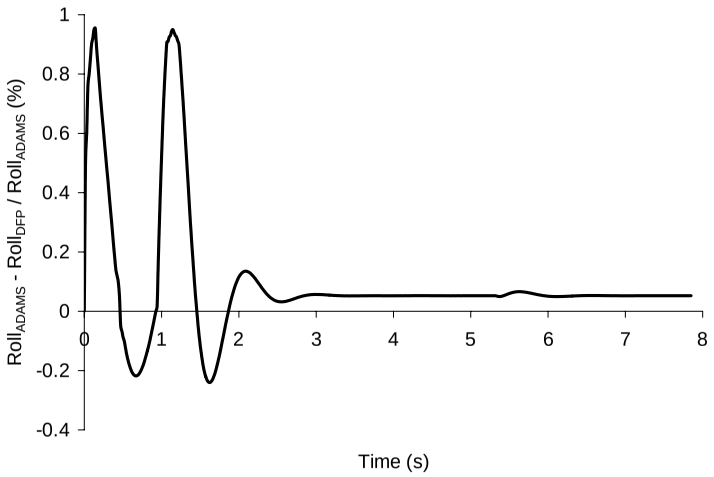
(a) Yaw Angle of the Vehicle Body



(b) Percent Difference in Calculating the Yaw Angle

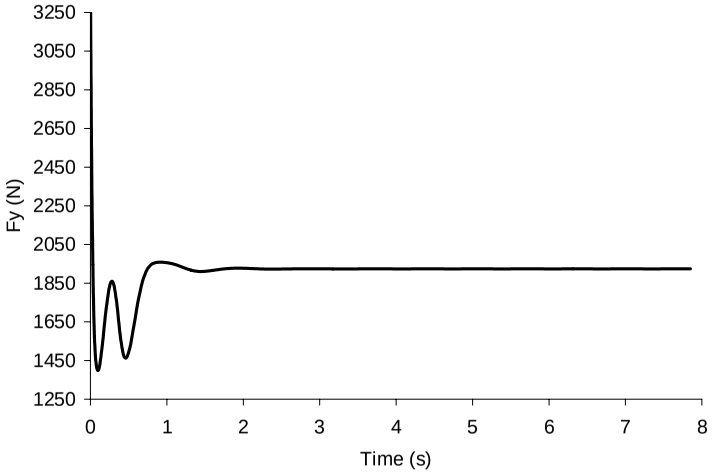


(c) Roll Angle of the Vehicle Body

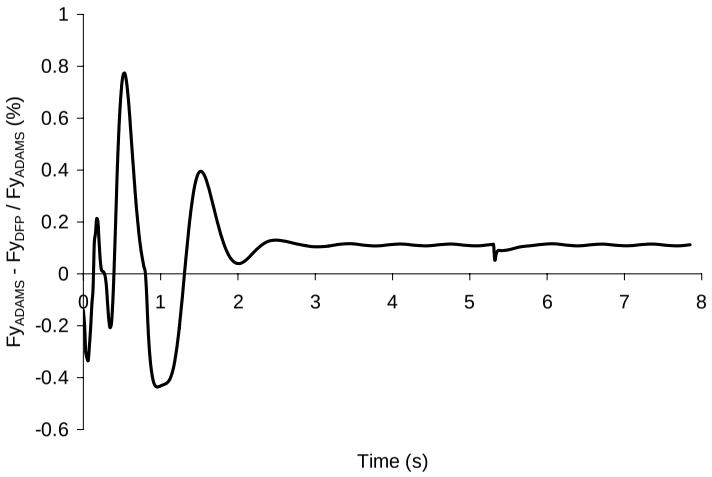


(d) Percent Difference in Calculating the Roll Angle

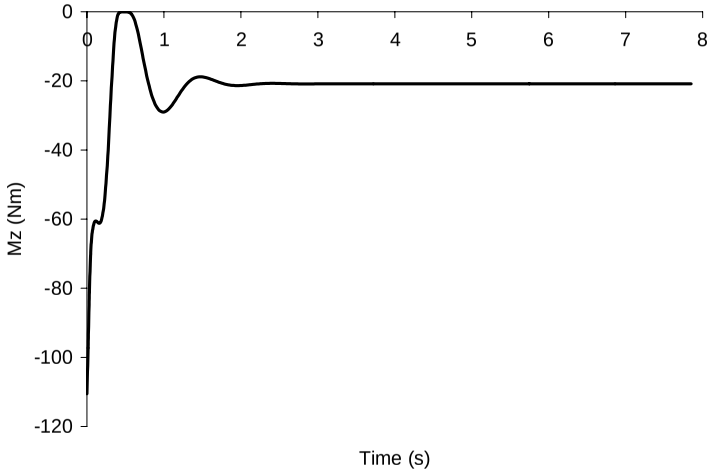
Figure 5.21: Response of the Chevrolet Equinox for the Example of a Constant Radius Cornering Maneuver



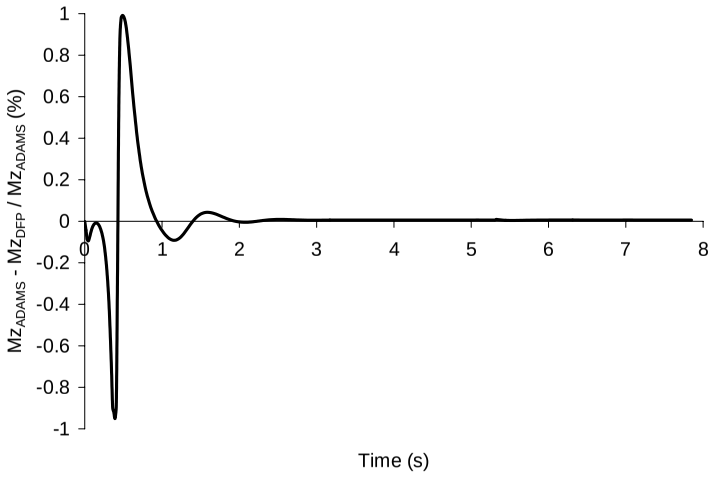
(a) Lateral Force of the Front Left (FL) Tire



(b) Percent Difference in Calculating F_y



(c) Aligning Moment of the FL Tire



(d) Percent Difference in Calculating M_z

Figure 5.22: Response of the Chevrolet Equinox for the Example of a Constant Radius Cornering Maneuver (Continued)

5.5.3 General Maneuver on a Pre-Determined Track

The last example involves the vehicle following a pre-defined track. This is a typical analysis to determine the vehicles performance as it follows a pre-defined path [23]. The pre-defined track for this example is a combination of the two road profiles that were used to model the road in the two previous examples (Sections (5.5.1) and (5.5.2)). Thus, the vehicle will first travel along a flat road and then up an inclined surface of 8.75% grade (5°) for some time before traveling on a flat surface once again (road surface is parallel to the XY plane, but is offset from it by a certain height). The vehicle will then perform a constant radius cornering maneuver on the elevated flat surface around a quarter of a circle that has a 50m radius.

The Chevrolet Equinox will perform this example with a constant forward velocity of 10m/s. Thus, a torque needs to be applied to the vehicle in order to overcome the rolling resistance moment and the gravitational effects that act to slow down the vehicle. Once again a proportional controller will be used to adjust the torque applied to each of the revolute joints appropriately to keep the velocity constant. The proportional controller shown in Equation (5.8) will be used with the same proportional constant, but with a desired velocity of 10m/s instead of 20m/s.

The road can not be defined in ADAMS/View for reasons discussed in Section (5.4.3). It is not possible to use the 2D road model to define the road because it does vary with respect to both the x and y directions (the road is a 3-D road). The 3D spline road model can not be used to model this example because with this model it is important that the vehicle follow the path defined by the centerline of the road and a driver model is not available in ADAMS/View. A driver model is required with this approach when the centerline of the road varies in the Z direction (excluding superimposed objects). Also, with the 3D spline road model it is important that the road remains smooth without any abrupt changes and in this particular example there is an abrupt change when the road changes from being flat to inclined and from inclined to flat again.

However, this road profile can be easily defined in DynaFlexPro with the tire/road component using two different methods. The road has to be defined in tabular form because it can not be defined by a single elevation function. It can be noted that the road can be specified as being independent of y since the road remains flat after the inclined surface, and can be modeled as $z = h$, where h is the height of the flat surface. Thus, the road can be modeled in tabular form by specifying x, z, zp_x, and α for each data point. Note, by doing this the road will be defined by 3 planes, $z = 0$, $z = 0.0875(x-l)$, and $z = h$, where l is the location when the surface changes from being flat to being inclined at a grade of 8.75%. For this example, let l be 100m and h be 5.25m; thus the road surface will go from being inclined to flat again at $x = 160$m (160m = 100m + (0.0875/5.25)m). Therefore, the complete

road surface can be defined by Equation (5.10).

$$z = \begin{cases} & x & < 100 \text{ m}: & 0 \text{ m} \\ 100 \text{ m} \leq & x & < 160 \text{ m}: & 0.0875(x - 100) \text{ m} \\ & x & \geq 160 \text{ m}: & 5.25 \text{ m} \end{cases} \tag{5.10}$$

Note, this equation is used to construct the data points of x, z, zp_x, and α that represent the road. The table of data can be found in Appendix (D). Let DFP1 indicate that the results were obtained from DynaFlexPro when the road is defined using this table.

The road can also be defined in DFP by specifying the centerline of the road in tabular form. In this particular fashion to define the road, the distance traveled along the path is used as the independent variable in the piecewise cubic splines that are used to represent each section of the road. The road is defined by specifying x, y, z, zp_s, and α for each data point. The tabular set of data used to represent the road is shown in Appendix (D). Let DFP2 indicate that the analysis is performed in DynaFlexPro with the centerline of the road being defined in tabular form. Thus, the Chevrolet Equinox is analyzed on this road trajectory in DynaFlexPro using both of the methods to define the road in order to validate the case of defining the centerline of the road in tabular form when the elevation of the road varies.

The steering angle applied to each of the motion drivers is used to ensure that the vehicle follows the path defined by the centerline of the road. The steering angle is initially set to 0 radians to force the vehicle to stay in a straight line. Once the constant radius cornering portion of the track is reached, the steering angle is changed to a constant to force the vehicle to follow the path and perform the cornering maneuver. The steering input is shown in Figure (5.23) and in Equation (5.11).

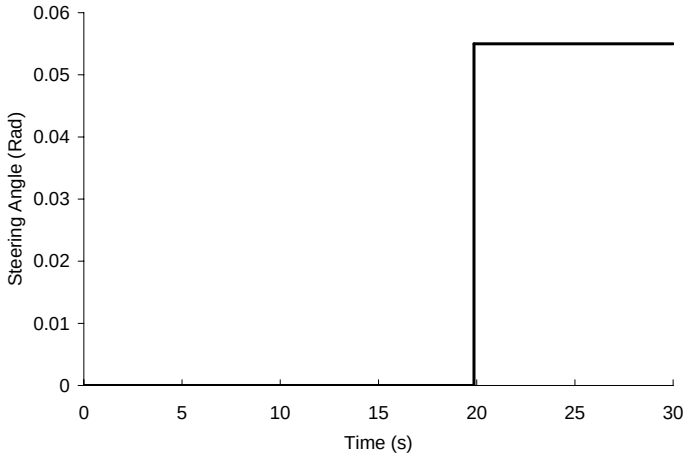


Figure 5.23: The Steering Motion Applied to Both Motion Drivers to Force the Vehicle to Follow the Centerline of the Road

$$\text{steer(t)} = \begin{cases} t & < 19.86s & 0 \text{ rad} \\ t & \geq 19.86s & 0.055 \text{ rad} \end{cases} \tag{5.11}$$

The analysis is performed in DynaFlexPro for a simulation duration of 27.85s for both methods to define the road. The time required to perform this analysis is easily determined because the velocity of the vehicle is held constant. The results from the example are shown in Figures (5.24)-(5.27) and in Table (5.6). It can be

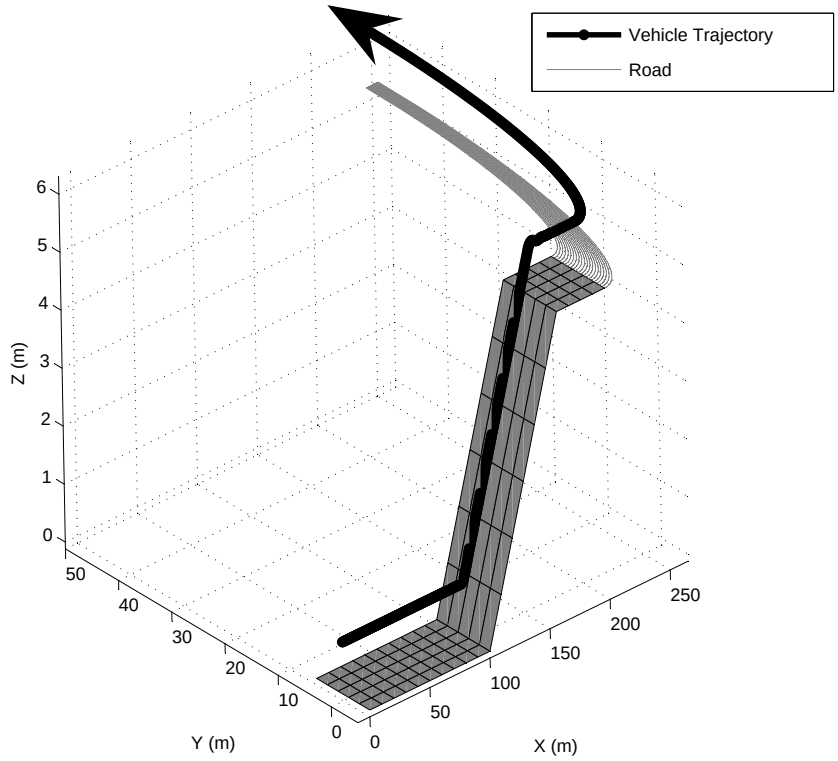


Figure 5.24: Vehicle Trajectory for the Example of the Chevrolet Equinox Following a Pre-Defined Track

Software	Simulation Time (s)
DFP1/Maple	81.0
DFP2/Maple	106.6

Table 5.6: The Simulation Times for the Example of the Chevrolet Equinox Following a Pre-Defined Track

seen from the results of the vehicle trajectory that the vehicle does indeed follow the path defined by the centerline of the road. This is most notable in the result of the top view of the vehicle trajectory. Thus, the steering angle is appropriate to keep the vehicle on the desired path and can be the parameter to be adjusted by a controller to keep the vehicle on the proper path. The result of the yaw angle confirms that the vehicle stays straight until the steering angle changes from 0 to 0.055 radians. The yaw angle also shows that the vehicle completes a 90° turn

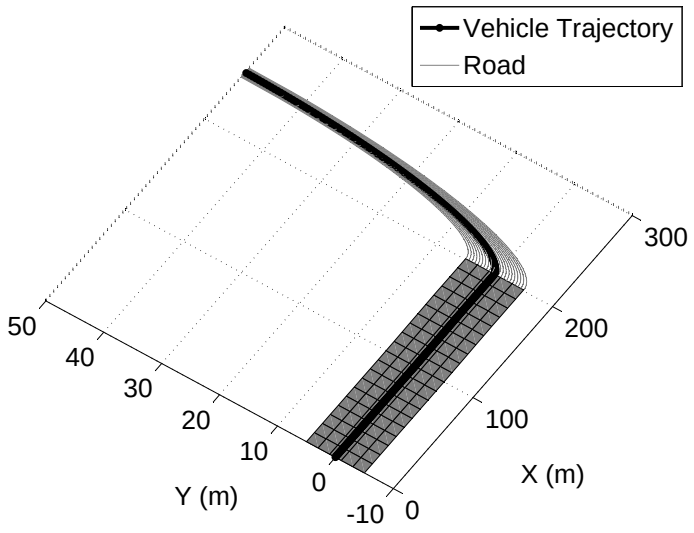


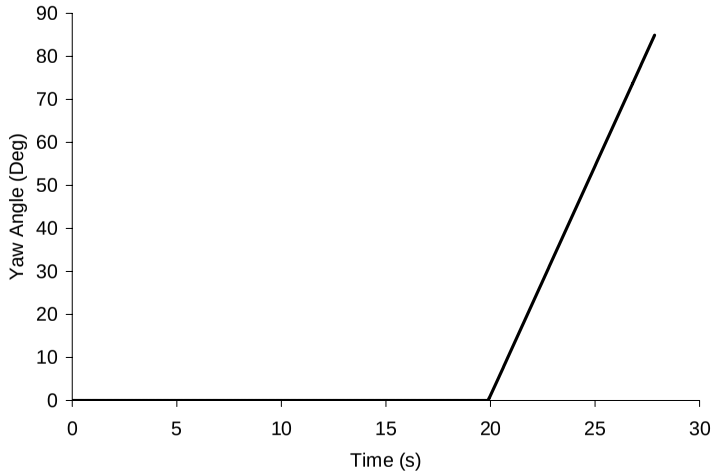
Figure 5.25: Top View of Vehicle Trajectory for the Example of the Chevrolet Equinox Following a Pre-Defined Track

during the last portion of the track. The roll angle response confirms that there are no lateral forces present until the vehicle starts to corner. This can also be seen in the response for the lateral force and aligning moment. It can further be seen in their responses that the steering angle is held constant during the cornering portion because their values are constant during that portion.

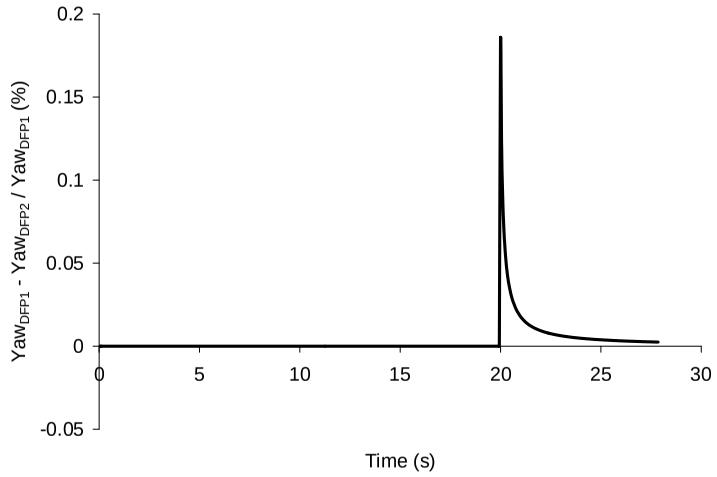
The % difference curves shown in Figures (5.26) and (5.27) show that the results obtained from DynaFlexPro for both methods to represent the road are in excellent agreement. Thus, this example confirms that DynaFlexPro with the tire/road component can be used to analyze vehicles on a pre-defined track by specifying the centerline of the road in tabular form.

The simulation is 1.3 times faster in DynaFlexPro when the road is modeled in tabular form as a function that only varies with respect to x versus when the centerline of the road is defined in tabular form. Thus, if simulation time is an important factor in analyzing the vehicle then it is important to use the most simplified road model that completely defines the road.

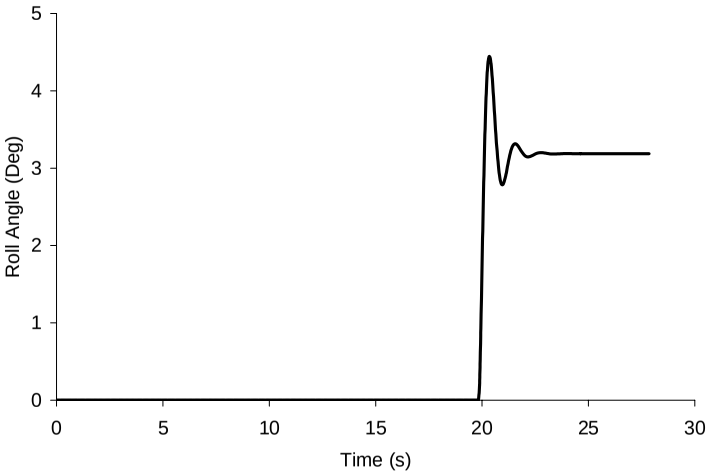
All 6 examples have shown the capabilities of the tire/road component to analyzing wheeled vehicle systems on 3-D roads. The examples have also shown that a significant reduction in the simulation time can be achieved when vehicle systems rolling on 3-D roads are analyzed in a symbolic environment rather than a numerical environment.



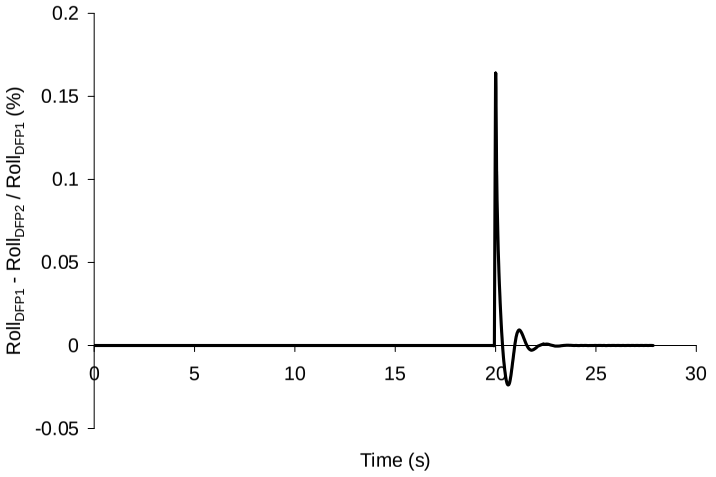
(a) Yaw Angle of the Vehicle Body



(b) Percent Difference in Calculating the Yaw Angle

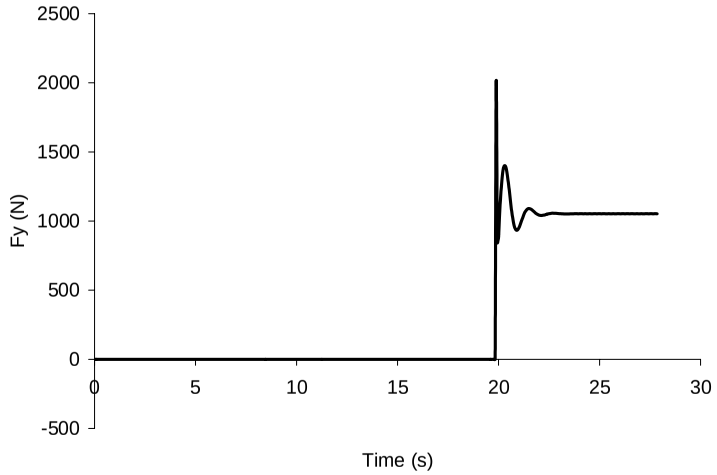


(c) Roll Angle of the Vehicle Body

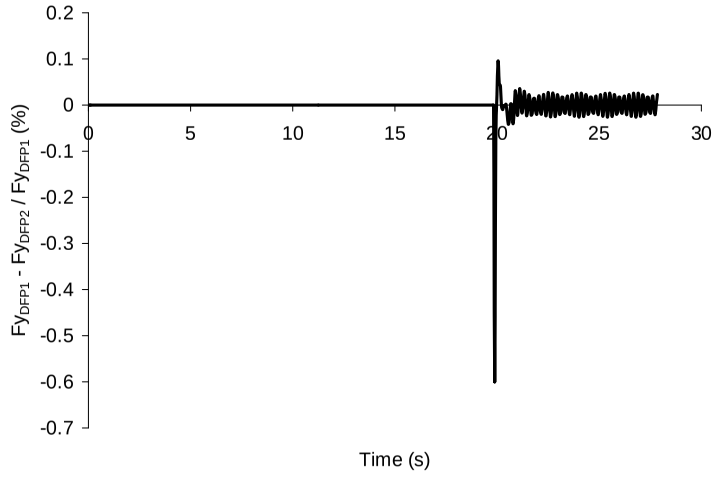


(d) Percent Difference in Calculating the Roll Angle

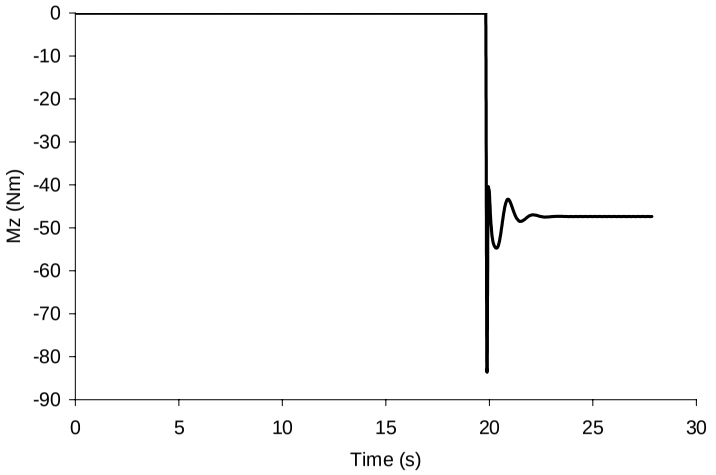
Figure 5.26: Response of the Example of the Chevrolet Equinox Following a Pre-Defined Track



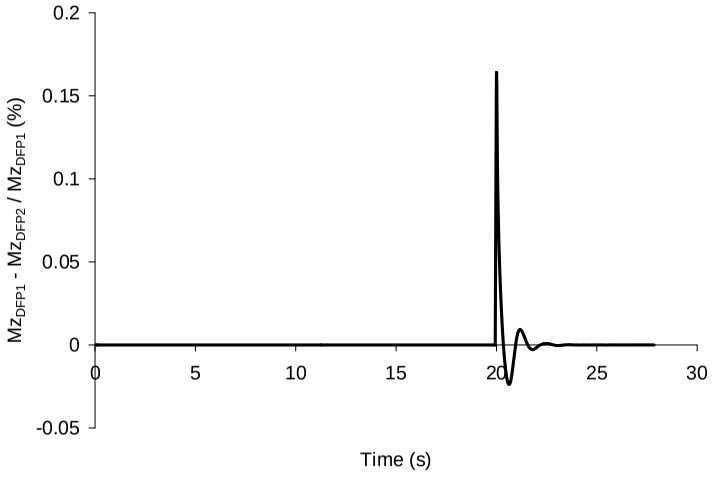
(a) Lateral Force of the Front Left (FL) Tire



(b) Percent Difference in Calculating F_y



(c) Aligning Moment of the FL Tire



(d) Percent Difference in Calculating M_z

Figure 5.27: Response of the Example of the Chevrolet Equinox Following a Pre-Defined Track (Continued)

Chapter 6

Conclusions and Future Work

6.1 Summary of the Current Work

The purpose of this research has been to develop an approach to analyze vehicle systems on 3-D roads while taking advantage of the promises offered by a symbolic formulation method. The goals were accomplished by developing the theory and creating a simulation code structure around the theory that permits the tire forces and moments to be evaluated in a systematic fashion for each tire in the vehicle system. The contact point is determined using the thin disk tire model with variable radius approach by generating and solving two non-linear algebraic equations for the loaded radius and its location using the Newton-Raphson iteration routine. The tire intermediate variables that are required as input by the tire model are calculated based on the current position and orientation of the tire and the location of the contact point. The final step of the simulation code structure involves the calculation of the tire forces and moments using a tire model. The structure allows the governing equations to be formulated with the tire forces and moments left as dummy variables that are evaluated during simulation.

The simulation code structure/road model procedure offers the following advantages.

- The structure is not limited to the symbolic formulation approach that is used to formulate the governing equations of the vehicle system. It can be implemented in any symbolic approach with the tire forces and moments left as dummy variables in the generalized force matrix. However, some work may be required to implement the structure in a software package as was shown with the development of tire/road component in order to realize the approach in DynaFlexPro.
- The approach is implemented in Maple which is a general purpose computer algebra software package. Thus, symbolic simplification tools in Maple can be used to reduce the computational cost associated with evaluating expressions,

and optimization methods can be used to optimize each of the procedures used in the structure.

- The road model procedure is automatically generated and optimized during the formulation stage using the single elevation function to define the road. This eliminates the road model from having to be hard-coded each time the function changes and thus allows different road profiles to be analyzed easily.

- The method permits the road to be defined in three different fashions depending on the known information of the road and its complexity. The approach was generated by assuming that the road can be defined by a single elevation function. However, if it can not be defined by a single elevation function then two tabular methods can be used to represent the road. The first represents the road as a plane that rotates in the cartesian space with respect to the Global Y axis only and the second as a set of data points that define the elevation of the centerline of the road.

- The Fiala tire model was used in the examples to calculate the tire forces and moments, but any tire model that is designed for handling analysis can be used. The approach is not dependent on the tire model. Also, the set of tire intermediate variables can be easily updated if any additional variable is needed.

- The method for constructing the procedures necessary to calculate the tire forces and moments permits the analysts to define his/her own external function that is necessary in the calculation of the forces and moments.

The structure was implemented in the DynaFlexPro software package by creating a linear graph representation of the tire and the road. The capability of using the simulation code structure to analyze the Chevrolet Equinox on different road profiles was shown through a set of examples. They showed the versatility of the approach and how the road can be defined using any one of the three methods presented. The Equinox was modeled using the open loop topology model outlined by Sayers [81] using a set of modeling variables that permitted only ordinary differential equations (ODEs) to be obtained. An MSC.ADAMS/View model was also created in order to validate the simulation code structure and to offer a comparison between a symbolical and a numerical formulation approach.

The results from the examples showed excellent agreement between those obtained from DynaFlexPro (DFP) and from MSC.ADAMS. They also showed the difficulties in defining a road in MSC.ADAMS and how it was impossible to analyze two out of the six roads considered because a driver model was not used. On the other hand, the road for each case could be implemented in DFP using the road model procedure. The DFP examples simulated 10 times faster than in MSC.ADAMS when the road was defined as a single elevation function and 4 times faster when the road was represented in tabular form. Thus, the simulation code structure can be used to analyze vehicle systems on 3-D roads.

6.2 Suggestions for Future Work

The simulation code structure developed to analyze wheeled vehicle systems on 3-D roads worked well for the examples presented in this thesis, but this approach does have its limitations. The main limitation is that the tire can have only one point of contact with the ground and in certain conditions this may not be true. Therefore, the thin disk tire model with variable radius approach should be extended to include methods to determine the proper location for force application when there is more than one point of contact with the ground. This could be accomplished using a weighted average of the contact points based on the tire deflection at each point or using a more complex approach such as the 3D equivalent volume model when more than one point of contact exists.

Another limitation to the approach is that the road is assumed to be rigid. Therefore, the model is not feasible to analyze vehicle systems on soft soil or deformable terrain. Thus, the road model and the method to determine the contact point should be extended to incorporate deformable surfaces. This can be accomplished using finite element methods to represent the road and the tire or by using an appropriate method to determine the contact area between the tire and the road when both are free to deform. The latter approach may be more computationally efficient than finite element methods and thus should be explored first.

The cubic spline is used to model the road between data points when it is represented in tabular form, and this spline may not be the most efficient. The distance traveled along the path is determined assuming the road is defined as a straight line between two data points. This may lead to the tire rolling on a false section of the road since the distance traveled along the path is not equivalent to the distance traveled along a straight line, and the error may accumulate. This will occur if the cubic spline function causes the road section to vary from the straight line. Other spline orders should be explored and studied in order to choose the desired one to represent the road.

The vehicle model with the proposed simulation code structure is not real-time applicable as is seen from the results obtained for the simulation time required for each simulation. Real-time applications are important in vehicle design since they are equipped with intelligent control strategies to improve the safety and performance of the vehicle [82]. These control strategies are designed using hardware-in-the-loop (HIL) simulations, where the actual system is replaced with a plant model that is used to represent the system. It is extremely important in HIL analysis that the simulations perform faster than real-time at each time step so that control strategies can be used to improve the performance of the vehicle. The simulation code structure includes an iterative approach that must be used to locate the contact point and this method is very close to real-time on an older PC. Thus, improvements to the Newton-Raphson iteration routine are needed to improve the computational cost of determining the contact point.

The Newton-Raphson iteration scheme can be improved by further optimizing

the road model procedure using optimization techniques. Also, the convergence criteria used in the iteration routine can be investigated to improve its efficiency. Monitoring the result of Equations (3.53) and (3.54) (the **F** matrix) may be more effective since this may cause the scheme to converge after only one iteration. Note, 2 iterations are always required when the results from the routine are monitored (Equations (3.55) and (3.56)). Additionally, the number of iterations may be decreased by investigating the necessity to check the conditioning of the determinant. It may be sufficient to use a non-spinning frame with a fixed initial guess without monitoring the determinant. Further improvements to the simulation time can also be achieved by compiling the code in c rather than in Maple's evalh environment; which is Maple's hardware float environment.

The simulation times are closer to real-time when the road is represented as a single elevation function than when represented in tabular form. This is because when the road is represented in tabular form there is an additional numerical iteration required for each tire to locate its current position inside the table that is used to define the road. Thus, improvements need to be made to both the procedure to determine the contact point and the procedure to locate the current position of the tire in the table.

The capabilities of the vehicle model can be improved by extending the tire model so that it is capable of performing durability analysis on the vehicle and components of the vehicle. This will also permit a frequency analysis to be performed on the vehicle to determine its natural frequencies. This can be accomplished by introducing tire models that are capable of predicting the behavior of the tire in a wide frequency range (up to 100Hz [62]). For low frequencies (up to 8Hz [74]), the tire primarily vibrates in a direction normal to the road surface and a single spring and damper is sufficient to represent the tire in this direction [74]. However, in higher frequencies the tire will also oscillate in the lateral and longitudinal directions and a single spring and damper along with a tire model for handling analysis is not sufficient to model the tire [62]. Tire models that are capable of predicting this behavior are ones based on finite element methods (FEM), the Flexible Ring Tire Model (FTire), and the Short Wavelength Intermediate Frequency Tyre Model (SWIFT) [62]. The incorporation of one of these tire models will involve a whole new contact model and methods to communicate between the tire model and the vehicle model.

One of the most common tests used to determine the ride quality of the vehicle is a four post test. In this test, the elevation of the road is varied with time at each tire in order to quantify the ride quality of the vehicle [23]. This test can be easily implemented by introducing a new tire model capable of properly determining the deflection and normal force of the tire at different road frequencies (such as the FTire model [62]) and by updating the elevation function used to represent the road to also be a function of time.

It is important that the vehicle follows the trajectory of the centerline of the road when it is defined in tabular form. In the examples presented in this thesis, the

steering angle input was used to ensure that the vehicle followed the trajectory of the road. However, a driver model should be developed that automatically adjusts the steering angle to force the vehicle to stay on the path defined by the road. It is also very common that a driver model is used when the road centerline is defined to force the vehicle to follow the trajectory [11, 10].

References

[1] *Iso 8855, road vehicles - vehicle dynamics and road holding ability - vocabulary.* International Organization for Standardization, 1991. 32, 33, 34, 35, 36, 38

[2] *CarSim Educational User Manual.* Mechanical Simulation Corporation, 2000. 59

[3] *Using ADAMS/Tire Road Models.* MSC.Software Corporation, 2003. 48, 59, 95, 101, 106

[4] *ADAMS/Car Manual.* MSC.Software Corporation, 2005. 23

[5] *ADAMS/Car RealTime Manual.* MSC.Software Corporation, 2005. 3, 21, 23

[6] *ADAMS/Solver Manual.* MSC.Software Corporation, 2005. 13

[7] *ADAMS/Tire Manual.* MSC.Software Corporation, 2005. 35, 39

[8] *DynaFlexPro User's Manual.* Maplesoft, 2006. 19, 34, 65, 87, 91

[9] *ve-Dyna's User's Guide.* Tesis DYNAware, http://www.tesis.de, 2006. 21, 23

[10] *Version 6 CarSim Math Model Features.* Mechanical Simulation Corporation, 2006. 27, 133

[11] *ADAMS/3D Road Manual.* MSC.Software Corporation, 2007. 26, 27, 49, 106, 133

[12] *Maple's User's Manual.* Maplesoft, 2007. 58, 65, 66, 71, 73, 97

[13] *Modeling Multibody Systems with ROBOTRAN.* Center for Research in Mechatronics, Belgium, 2008. 17

[14] *2010 Chevrolet Equinox Brochure.* General Motors of Canada Limited, 2009. 90

[15] *MapleSim 2 User's Manual.* MapleSoft Inc., 2009. 19

[16] K. Atkinson. Modelling a road using spline interpolation. *Reports of Computational Mathematics #145, Department of Mathematics, University of Iowa, Iowa City, USA*, 2002. 59, 61

[17] N.L. Azad, J. McPhee, and A. Khajepour. Off-road lateral stability analysis of an articulated steer vehicle with rear-mounted load. *International Journal of Vehicle Systems Modelling and Testing*, 1(1), 2005. 24

[18] E. Bakker, L. Nyborg, and H.B. Pacejka. Tyre modelling for use in vehicle dynamics studies. *SAE Technical Paper Series*, paper no. 870421, 1986. 43

[19] H. Baruh. *Analytical dynamics*. The McGraw-Hill Companies, Inc., United States, 1999. 8, 9, 12, 13, 14

[20] J. Baumgarte. Stabilization of constraints and integrals of motion. *Computer Methods in Applied Mechanics*, 1:1–16, 1972. 12

[21] M. G. Bekker. Off-the-road locomotion. *University of Michigan Press*, 1960. 24

[22] J.E. Bernard and C.L. Clover. Tire modelling for low-speed and high-speed calculations. *SAE Technical Paper Series*, paper no. 950311, 1995. 39

[23] M. Blundell and D. Harty. *The Multibody Systems Approach to Vehicle Dynamics*. Society of Automotive Engineers, Inc., Warrendale, Pennsylvania, 2004. 21, 24, 25, 26, 27, 29, 30, 31, 38, 40, 41, 43, 44, 48, 49, 64, 83, 87, 88, 90, 123, 132

[24] W. Bombardier and J. McPhee. Symbolic modeling of wheeled vehicles on 3d roads. *Multibody Dynamics 2009 Conference*, Warsaw, June 28^{th}-July 2^{nd}, 2009. 15, 28, 57, 87

[25] M. Borri, C. Bottasso, and P. Mantegazza. Equivalence of kane's and maggie's equations. *Journal of Meccania, Springer Netherlands*, 25(4):272–274, 1990. 17

[26] E. Carpanzano and C. Maffezzoni. Symbolic manipulation techniques for model simplification in object-oriented modelling of large scale continuous systems. *Mathematics and Computers in Simulation*, 48(2):133–150, 1998. 18

[27] M. Castro, L. Iglesias, R. Rodriguez-Solano, and J.A. Sanchez. Geometric modelling of highways using global positioning system (gps) data and spline approximation. *Transport Research Part C*, 14:233–243, 2006. 59

[28] A. M. Cohen, X-S Gao, and N. Takayama. *Mathematical software, proceedings of the first international congress of mathematical software*. World Scientific Publishing Co. Pte. Ltd., River Edge, New Jersey, 2002. 71

[29] D. Cox, J. Little, and D. O'Shea. *Using Algebraic Geometry*. Springer-Verlag New York, Inc., New York, New York, 1998. 66

[30] J. Cuadrado, J. Cardenal, and E. Bayo. Modeling and solution methods for efficient real-time simulation of multibody dynamics. *Multibody System Dynamics*, 1(3):259–280, 1997. 13

[31] L. D'Amato. `UW` gets `$1M` to design better vehicles. *The Record (therecord.com), June 30*, 2009. 1

[32] D. C. Davis. A radial-spring terrain-enveloping tire model. *Vehicle System Dynamics*, 4 (1):55–69, 1975. 50

[33] J. C. Dixon. *Tires, suspension and handling, second edition*. Society of Automotive Engineers, Inc., Warrendale, Pennsylvania, 1996. 51

[34] Green Car Journal Editors. *Chevrolet Equinox Fuel Cell Wins 2008 Green Vision Award*. GreenCar.com (January 28^{th}, 2008), Demand Media, Inc., 2008. 90

[35] H. Elmqvist, F. E. Cellier, and M. Otter. Object-oriented modeling of hybrid systems. *Proceedings of the 1993 European Simulation Symposium*, 1993. 18

[36] H. Elmqvist, S. E. Mattsson, H. Olsson, J. Andreasson, M. Otter, C. Schweiger, and D. Brück. Realtime simulation of detailed vehicle and powertrain dynamics. *SAE Technical Paper Series*, (2004-01-0768), 2004. 18, 23, 24

[37] H. Elmqvist and H. Olsson. New methods for hardware-in-the-loop simulation of stiff models. *In Proceedings of the 2nd International Modelica Conference*, (2004-01-0768):59–64, 2002. 18

[38] E. Fiala. Seitenkrafte am rollenden luftreifen. *VDI-Zeitschrift*, 96:973, 1954. 26, 41

[39] P. Fisette, T. Postiau, L. Sass, and J.C. Samin. Fully symbolic generation of complex multibody models. *Mechanics of Structures and Machines*, 30(1):31–82, 2002. 4, 12, 13, 16, 17

[40] P. Fisette and J.C. Samin. A new wheel/rail contact model for independent wheels. *Archive of Applied Mech.*, 64:180–219, 1994. 55

[41] T. Geike and J. McPhee. Inverse dynamic analysis of parallel manipulators with full mobility. *Mechanism and Machine Theory*, 38:549–562, 2003. 68

[42] G. Genta. *Motor Vehicle Dynamics Modeling and Simulation*. World Scientific Publishing Co. Pte., Ltd., Singapore, 2003. 50

[43] T.D. Gillespie. *Fundamentals of Vehicle Dynamics*. Society of Automotive Engineers, Inc., Warrendale, Pennsylvania, 1992. 25, 32, 69, 86, 105

[44] J. H. Ginsberg. *Advanced engineering dynamics (second edition)*. Cambridge University Press, New York, New York, 2007. 8, 9, 12

[45] H. Gontran, P.-Y. Gilliéron, and J. Skaloud. Precise road geometry for integrated transport safety systems. *5th Swiss Transport Research Conference*, 2005. 59

[46] D. V. Griffiths and I. M. Smith. *Numerical Methods for Engineers*. Taylor and Francis Group, LLC, Boca Raton, Florida, 2006. 60, 69

[47] E. J. Haug. *Computer Aided Kinematics and Dynamics of Mechanical Systems*. Allyn and Bacon, Needham Heights, MA, 1989. 9, 10, 11, 12, 14, 16

[48] E. J. Haug, D. Negrut, R. Serban, and D. Solis. Numerical methods for high-speed vehicle dynamic simulation. *Mechanics Based Design of Structures and Machines*, 27(4):507–533, 1999. 3, 16

[49] Y. He, A. Khajepour, J. McPhee, and X. Wang. Dynamic modelling and stability analysis of articulated frame steer vehicles. *International Journal of Heavy Vehicle Systems*, 12(1):28–59, 2005. 21

[50] P. Hewson. Method for estimating tyre cornering stiffness from basic tyre information. *Procedures of the Institute of Mechanical Engineers, Part E: J Automobile Engineering*, 219:paper no. D17604, 2005. 42

[51] R. C. Hibbeler. *Engineering Mechanics: Dynamics (12th edition)*. Pearson Education, Inc., Upper Saddle River, New Jersey, 2009. 7

[52] S. Horiuchi, K. Okada, and S. Nohtomi. Improvement of vehicle handling by nonlinear integrated control of four wheel steering and four wheel torque. *Journal of the Society of Automotive Engineers, Japan*, 20:459–464, 1999. 100, 101

[53] K. H. Huebner. *The Finite Element Method for Engineers*. John Wiley & Sons, Inc., New York, New York, 2001. 69

[54] J. G. Jalón and E. Bayo. *Kinematic and Dynamic Simulation of Multibody Systems: The Real-Time Challenge*. Springer-Verlag, New York, 1994. 14

[55] D.J. Jeffrey. High precision computation of elementary functions in maple. *Proceedings of Computer Algebra in Scientific Computation 2002*, pages 183–190, 2002. 67, 71

[56] A. Kecskemethy, T. krupp, and M. Hiller. Symbolic processing of multiloop mechanism dynamics using closed-form kinematic solutions. *Multibody System Dynamics*, 1(1):23–45, 1997. 13

[57] J. H. Kim, K. S. Kim, and Y. J. Kang. Ride comfort evaluation and suspension design using axiomatic design. *Journal of Mechanical Science and Technology*, 21:1066–1076, 2007. 16

[58] E. Klum. *Chevy Equinox Receives Green Award*. ArticleBase.com, January 24^{th}, 2008, 2008. 90

[59] M. Leger and J. McPhee. Selection of modeling coordinates for forward dynamic multibody simulations. *Multibody System Dynamics*, 18(2):277–297, 2007. 19

[60] Z. Li and S. Kota. Application of multi-body computer codes to road vehicle dynamics modelling problems. *Journal of Computing and Information Science in Engineering*, 1:276–279, 2001. 15

[61] Z. Liu, G. Payre, and P. Bourassa. Nonlinear oscillations and chaotic motions in a road vehicle system with driver steering control. *Nonlinear Dynamics*, 9:281–304, 1994. 64

[62] P. Lugner, H. Pacejka, and M. Plöchl. Recent advances in tyre models and testing procedures. *Vehicle System Dynamics*, 43(6):413–426, 2005. 55, 132

[63] S. Mammar and D. Koenig. Vehicle handling improvement by active steering. *Vehicle System Dynamics*, 38(3):211–242, 2002. 100

[64] J. McPhee. Unified modelling theories for the dynamics of multidisciplinary multibody systems. *Advances in Computational Multibody Systems*, 129-158, 1996. 19

[65] J. McPhee. Automatic generation of motion equations for planar mechanical systems using the new set of "branch coordinates". *Mechanism and Machine Theory*, 35:805–823, 1998. 8, 14, 19, 20, 21, 87, 88, 94, 145, 147, 154

[66] J. McPhee and S. Redmond. Modelling multibody systems with indirect coordinates. *Computer Methods in Applied Mechanics and Engineering*, in press, 2005. 14, 19, 21

[67] J. McPhee, C. Schmitke, and S. Redmond. Dynamic modelling of mechatronic multibody systems with symbolic computing and linear graph theory. *Mathematical and Computer Modelling of Dynamic Systems*, 10(1):17–23, 2004. 87

[68] W. F. Milliken and D. L. Milliken. *Race Car Vehicle Dynamics*. Society of Automotive Engineers, Inc., Warrendale, Pennsylvania, 1995. 40, 117

[69] P. Mitiguy and T. Kane. Motion variables leading to efficient equations of motion. *International Journal of Robotics Research*, 15(5):522–532, 1996. 14

[70] K. Morency. *Automatic Generation of Real-Time Simulation Code for Vehicle Dynamics using Linear Graph Theory and Symbolic Computing*. MASc thesis, University of Waterloo, 2007. xii, 4, 21, 22, 24, 73, 80, 82, 88, 89, 94

[71] R. A. Nocolaides and N. Walkington. *Maple: A Comprehensive Introduction*. Cambridge University Press, New York, New York, 1996. 73

[72] U. Nordström, J. Diaz Lopez, and H. Elmqvist. Automatic fixed-point code generation for modelica using dymola. *The Modelica Association*, September 4 - 5, 2006. 18, 67

[73] C. Oertel and A. Fandre. Ride comfort simulations and steps towards life time calculations: Rmod-k and adams. *International ADAMS User Conference*, Berlin, 1999. 48, 49

[74] H.B. Pacejka. *Tire and Vehicle Dynamics (2nd Edition)*. Society of Automotive Engineers, Inc., Warrendale, Pennsylvania, 2006. 25, 26, 27, 30, 31, 35, 38, 39, 40, 44, 45, 132

[75] H.B. Pacejka and R.S. Sharp. Shear force development by pneumatic tyres in steady state conditions: a review of modelling aspects. *Vehicle System Dynamics*, 20 (3-4):121–176, 1991. 26, 38, 41

[76] T. Postiau, L. Sass, P. Fisette, and J. C. Samin. High-performance multibody models of road vehicles: Fully symbolic implementation and parallel computation. *20th International Congress of Theoretical and Applied Mechanics*, pages 57–83, 2000. 4, 23, 26, 27, 29, 30, 33, 37, 48, 55, 56

[77] W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery. *Numerical Recipes in C: the Art of Scientific Computing (2nd edition)*. Cambridge University Press, New York, New York, 1992. 58, 61, 85, 95

[78] Rajamani R. *Vehicle Dynamics and Control*. Springer Science and Business Media, Inc., New York, New York, 2006. 59

[79] J. Ryu and J. C. Gerdes. Integrating inertial sensors with global positioning system (gps) for vehicle dynamics control. *ASME Journal of Dynamic Systems, Measurement, and Control*, 126:243–254, 2004. 51

[80] L. Sass. *Symbolic Modeling of Electromechanical Multibody Systems*. Phd thesis, Université Catholique de Louvain, 2004. 17

[81] M. W. Sayers and D. Han. Generic multibody vehicle model for simulating handling and braking. *Vehicle System Dynamics*, 25 Suppl:599–613, 1996. 1, 3, 4, 21, 22, 23, 24, 28, 29, 30, 38, 39, 130

[82] M.W. Sayers. Vehicle models for rts applications. *Vehicle System Dynamics*, 32(4):421–438, 1999. 2, 3, 21, 22, 23, 91, 131

[83] M.W. Sayers and A. Stribersky. Computer algebra in nonlinear analyses of the straightline stability of combination vehicles. *Computers and Structures*, 44(1-2):179–186, 1992. 9, 18

[84] W. E. Schiehlen. *Multibody Systems Handbook*. Springer-Verlag, New York, 1990. 15

[85] I. C. Schmid. Interaction of vehicle and terrain results from 10 years research at ikk. *Journal of Terramechanics*, 32(1):3–26, 1995. 26

[86] I. C. Schmid, C. Harnish, and B. Lach. Terrain-vehicle interactions in virtual reality. *Seoul 2000 FISITA World Automotive Congress, Seoul, Korea, June, 12-15, 2000*, (F2000IS006), 2000. 26

[87] C. Schmitke. *Modelling Multibody Multi-Domain Systems Using Subsystems and Linear Graph Theory*. Phd thesis, University of Waterloo, 2004. 89

[88] C. Schmitke and J. McPhee. Forming equivalent subsystem components to facilitate the modelling of mechatronic multibody systems. *Multibody System Dynamics*, 14:81–109, 2005. 9, 20, 21, 88, 89

[89] C. Schmitke, K. Morency, and J. McPhee. Using graph theory and symbolic computing to generate efficient models for multi-body vehicle dynamics. *Proceedings of the Institution of Mechanical Engineers, Part K: Journal of Multi-body Dynamics*, 222:339–352, 2008. 2, 3, 4, 21, 33, 87

[90] K. Schmitt and J. Madsen. A gaussian process based approach for handling uncertainty in vehicle dynamics simulation. *Proceedings of the 2008 ASME International Mechanical Engineering Congress and Exposition, Boston, MA*, 2008. 106

[91] R. S. Sharp. Application of multi-body computer codes to road vehicle dynamics modelling problems. *Proceedings of the Institute of Mechanical Engineers, Part D: Journal of Automobile Engineering*, 208(1):55–61, 1994. 14

[92] J. Stewart. *Calculus Early Transcendentals (5^{th} Edition)*. Thomson Learning, Inc., Belmont, California, 2003. 32

[93] J. S. Sui and J. A. Hirshey. Analytical tire model for vehicle durability and ride comfort analysis. U.S. Patent no. 6741957, 2004. 51

[94] H. E. Tseng. Dynamic estimation of road bank angle. *Vehicle System Dynamics*, 36:307–328, 2001. 61

[95] J.J.M. van Oosten. `TMPT` tire modeling in `ADAMS`. *Vehicle System Dynamics*, 45 (Suppl):191–198, 2007. 30

[96] J.J.M. van Oosten, H.-J. Unrau, A. Riedel, and A. Bakker. Standardization in tire modeling and tire testing — tydex workgroup, time project. *Tire Science and Technology*, 27(3):188–202, 1999. 29

[97] J. Villemonteix, E Vazquez, and E Walter. An informational approach to the global optimization of expensive-to-evaluate functions. *Journal of Global Optimization*, 44(4):509–534, 2009. 68

[98] D. Wang and F. Qi. *Trajectory planning for a four-wheel-steering vehicle.* Proceedings of the 2001 IEEE International Conference on Robotics and Automation, Seoul, Korea, May 21-26, 2001. 51

[99] H.J. Weber and G. B. Arfken. *Essential Mathematical Methods for Physicists.* Academic Press, San Diego, California, 2004. 68, 69

[100] D. T. Wetswick, K. George, and M. Verhaegen. *Nonlinear identification of automobile vibration systems.* Proceedings of the 7th Mediterranean Conference on Control and Automation (MED99), Haifa, Israel, June 28-30, 1999. 50

[101] A. B. Will and S. H. Zak. Modelling and control of an automated vehicle. *Vehicle System Dynamics*, 27(3):131–155, 1997. 100

[102] A. Wittkopf. Automatic code generation and optimization in maple. *Journal of Numerical Analysis, Industrial and Applied Mathematics*, 3(1-2):167–180, 2008. 3, 74, 75

[103] J. Y. Wong. *Theory of Ground Vehicles, Second Edition.* John Wiley & Sons, Inc., New York, New York, 1993. 24

Appendix A

Planar Manipulator Example

An example of how the Newton-Euler equations and linear graph theory can be used to obtain the equations of motion of multibody dynamic systems is presented in this section. The RR (revolute revolute) planar manipulator will be the multibody system used for example purposes, and is shown in Figure (A.1). The goal of

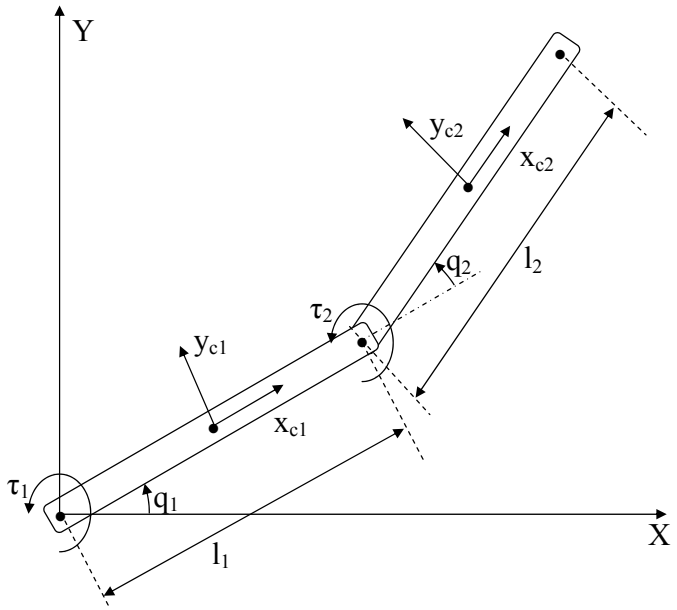


Figure A.1: Planar Manipulator

this example is to determine expressions for the motor torques (τ_1 and τ_2). These expressions are the equations of motion that are commonly used in robotics since the motor torques are of primary interest. For this example the gravitational field will act in the negative Y direction, and the links are assumed to be slender rods.

The solution to this example problem will be solved first using linear graph theory and then using the Newton-Euler equations and for both approaches the steps to solve the problem will be outlined.

A.1 Solution Using Linear Graph Theory (LGT)

Step 1: Determine the number of the degrees of freedom (DOF) of the system.

Since this problem is a 2-Dimensional (2-D) problem, Grübler's equation can be used to calculate the DOF (f) of the problem.

$$\begin{aligned} f &= 3nb - 2nj \\ nb &\Rightarrow \# \text{ of bodies} \\ nj &\Rightarrow \# \text{ of joints} \\ f &= 3 \cdot 2 - 2 \cdot 2 = 2 \end{aligned}$$

Step 2: Specify the generalized coordinates that will be used to solve the problem.

Since the objective of the problem is to determine the motor torques, a joint coordinate formulation will be used.

$$\mathbf{q} = \left\{ \begin{array}{c} q_1 \\ q_2 \end{array} \right\}$$

Step 3: Determine where the global (base frame) will be located.

The location of the global frame is shown in Figure (A.1).

Step 4: Determine the mass of each body and the location of the center of mass of each each body.

The bodies in the system in consideration are the links, and since the problem will be solved symbolically the mass of link 1 is m_1, and the mass of link 2 is m_2. The center of mass of each link is at the center of the link.

Step 5: Place a local frame on each center of mass (COM), and specify the x, y, and z directions of the frame.

The frames for each link are shown in Figure (A.1), where x_{c1} and y_{c1} is the frame of the first link, and x_{c2} and y_{c2} of the second.

Step 6: Determine the dyadic matrix ($\mathbf{I}$) of each body.

The inertia matrix of both links with respect to their local frame are as follows:

$$\mathbf{I_1} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & \frac{1}{12} m_1 l_1^2 & 0 \\ 0 & 0 & \frac{1}{12} m_1 l_1^2 \end{bmatrix} \quad \mathbf{I_2} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & \frac{1}{12} m_2 l_2^2 & 0 \\ 0 & 0 & \frac{1}{12} m_2 l_2^2 \end{bmatrix}$$

Step 7: Determine the rotation matrices for all bodies in the system which transform the frame of each body to the global frame. Also, determine the rotation matrices which relate the frame of one body to the frame of another for the bodies which are linked together through a joint.

The rotation matrices for the planar manipulator are as follows:

$$R_0^1 = \begin{bmatrix} \cos(q_1) & -\sin(q_1) & 0 \\ \sin(q_1) & \cos(q_1) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$R_0^2 = \begin{bmatrix} \cos(q_1+q_2) & -\sin(q_1+q_2) & 0 \\ \sin(q_1+q_2) & \cos(q_1+q_2) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$R_1^2 = \begin{bmatrix} \cos(q_2) & -\sin(q_2) & 0 \\ \sin(q_2) & \cos(q_2) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Where, R_b^a indicates that this matrix transforms a vector that is with respect to the a frame to the b frame.

Step 8: Determine the linear graph representation of the system. This graph is directly related to the topology of the system.

The linear graph for the planar manipulator is shown in Figure (A.2).

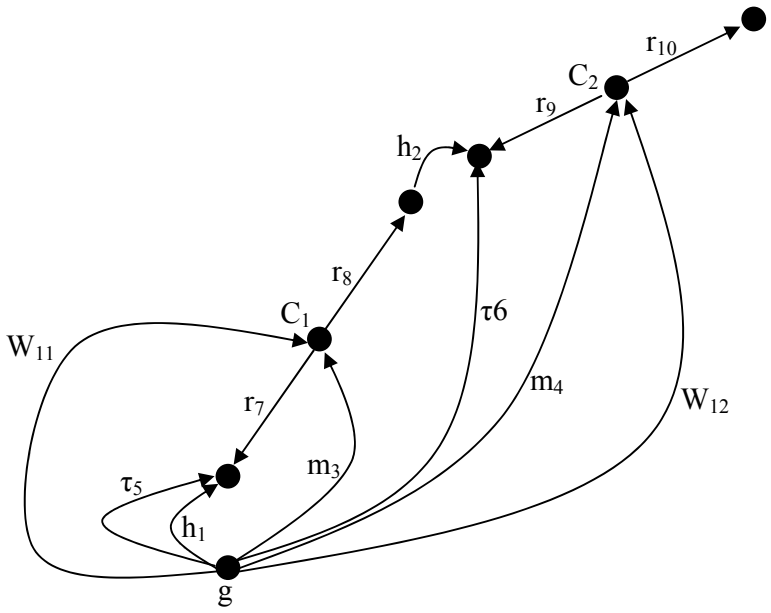


Figure A.2: Linear Graph Representation of the Planar Manipulator

Note that the physical system can be seen in the linear graph. Table (A.1) shows what each edge in the tree represents.

Step 9: Specify the translational tree (T-tree) and the rotational tree (R-tree) in the linear graph. The trees chosen for the problem will reflect the generalized coordinates chosen for the problem.

τ	External Torque
m	Rigid Body
r	Rigid Arm Element
W	External Force (Weight)
h	Revolute Joint

Table A.1: Labeling Used in the Linear Graph that Represents the Planar Manipulator

Since the set of generalized coordinates for this problem are the joint coordinates of the manipulator (q_1 and q_2), the two revolute joints will need to be selected into the R-tree. In order to only have the equations of motion in terms of the joint coordinates, the rigid arm elements will be selected into both trees and the revolute joints will also be selected into the T-tree since these joints have no motion in the translational domain. The R-tree and the T-tree for this particular problem are the same and is shown in Figure (A.3) as a single tree. Note the thicker edges are the edges in the tree and thus are the branches.

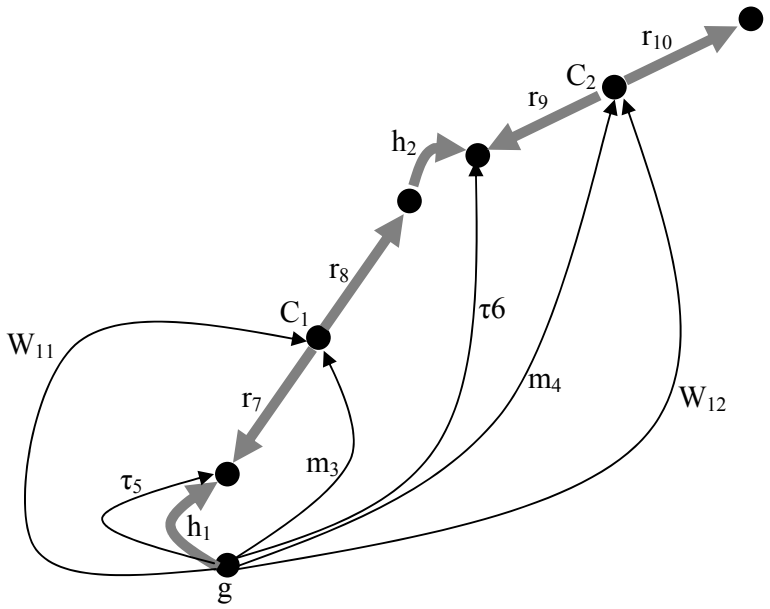


Figure A.3: The Translational and Rotational Tree of the Planar Manipulator

Step 10: Obtain the equations of motion of the system. This is accomplished by finding the fundamental cutest (f-cutset) equation for each of the components in the trees (R-tree and T-tree). The equations are then obtained by projecting the f-cutsets onto the motion space of each of the components. Recall, a f-cutset consists of one branch and a unique set of chords and divides the graph in exactly two parts [65].

The two f-cutsets are shown in Figure (A.4) for the two link planar manipulator (these are shown by the dashed gray lines). Since there are only two components in the T-tree and the R-tree with a non-zero motion space (the motion space of the rigid arm elements is zero), only two f-cutsets are needed: one for each revolute joint. Also note that only two equations will be obtained, since the motion space of a revolute joint is zero in the translational domain and is along the axis of rotation in the rotational domain.

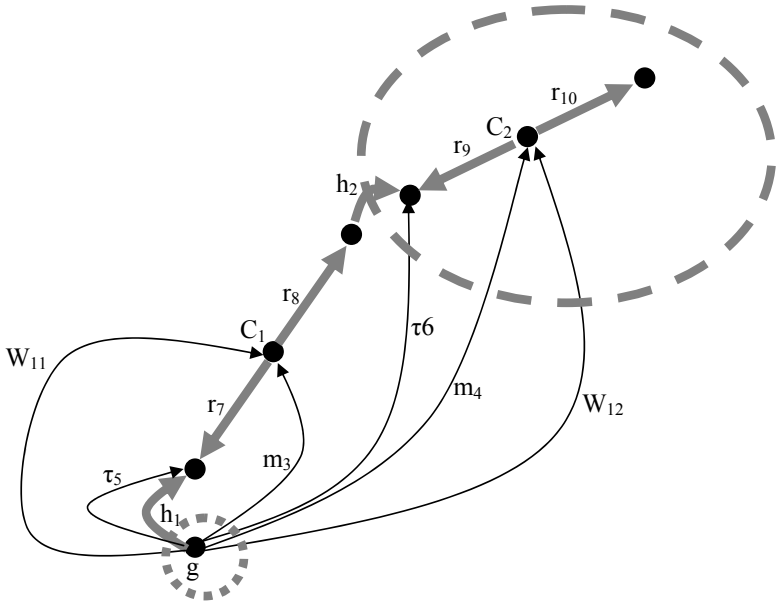


Figure A.4: The Fundamental Cutsets of the Planar Manipulator

The f-cutset for the first revolute joint (h_1) is as follows:

$$\vec{T}_{W11} + \vec{\tau}_5 + \vec{T}_{h1} + \vec{T}_{m3} + \vec{\tau}_6 + \vec{T}_{m4} + \vec{T}_{W12} = 0$$
$$\vec{\tau}_5 + \vec{T}_{m3} + \vec{\tau}_6 + \vec{T}_{m4} = 0$$

Note, the weight acts at the center of mass, and will thus not produce a torque about the center of mass; the revolute joints are assumed to be ideal and thus there will be no friction torques which act at these joints.

The f-cutset for the second revolute joint (h_2) is as follows:

$$\vec{T}_{h2} + \vec{\tau}_6 + \vec{T}_{m4} + \vec{T}_{W12} = 0$$
$$\vec{\tau}_6 + \vec{T}_{m4} = 0$$

The equations of motion are then obtained by projecting these f-cutset equations onto the motion space of the joint in the equation. The motion space of the first revolute joint is the z-axis of the first link, and the motion space of the second

revolute joint is the z-axis of link 2. Therefore the equations of motion are as follows.

$$\begin{matrix}\left(\vec{\tau}_5 + \vec{T}_{m3} + \vec{\tau}_6 + \vec{T}_{m4} = 0\right) \cdot \hat{k}_1 \\ \left(\vec{\tau}_6 + \vec{T}_{m4} = 0\right) \cdot \hat{k}_2\end{matrix}$$

Step 11: Substitute the terminal equations for all of the components in the equations of motion. The terminal equations represent the physical characteristic of each edge in the linear graph.

$$\begin{matrix}\left(\begin{array}{l} \tau_1\hat{k}_1 - \mathbf{I_1} \cdot \ddot{\vec{q}}_1 - \vec{r}_7 \times \vec{F}_7 - \vec{r}_8 \times \vec{F}_8 + \tau_2\hat{k}_1 \\ -\mathbf{I_2} \cdot \left(\ddot{\vec{q}}_2 + \ddot{\vec{q}}_1\right) - \vec{r}_9 \times \vec{F}_9 = 0 \end{array}\right) \cdot \hat{k}_1 \\ \left(\tau_2\hat{k}_2 - \mathbf{I_2} \cdot \left(\ddot{\vec{q}}_2 + \ddot{\vec{q}}_1\right) - \vec{r}_9 \times \vec{F}_9 = 0\right) \cdot \hat{k}_2\end{matrix}$$

Step 12: Eliminate the secondary variables from the equations of motion using the branch and chord transformations. Note that by doing this, the equations obtained will be in terms of the across variables of the edges in the tree and the through variables of the edges in the cotree. Therefore, the terminal equations need to be used to introduce the physical significance of the added components into the equations. The tree through variables and the cotree across variables are the secondary variables [65]. For a mechanical system the through variables are forces and torques, and the across variables are displacements, velocities, and accelerations. A chord transformation will allow the through variables in the tree to be specified as a function of the through variables in the cotree (a primary variable) [65]. This is accomplished by writing the f-cutset for each through variable in the tree. A branch transformation will allow the across variables of the cotree to be specified as a function of the across variables of the tree (primary variables) [65]. This is accomplished by writing the fundamental circuit (f-circuit) equations for each of the across variables in the cotree.

The secondary variables in the equations of motion of the 2 link planar manipulator are the three force vectors ($\vec{F}_7, \vec{F}_8, \vec{F}_9$). These are eliminated using the f-cutset of each of the rigid arm elements. Figure (A.5) outlines these cutsets (shown by the dashed lines).

The following equations are obtained from each of the cutsets in order to eliminate the secondary variables. Note that the external torques would not be in the T-tree, so they can be ignored.

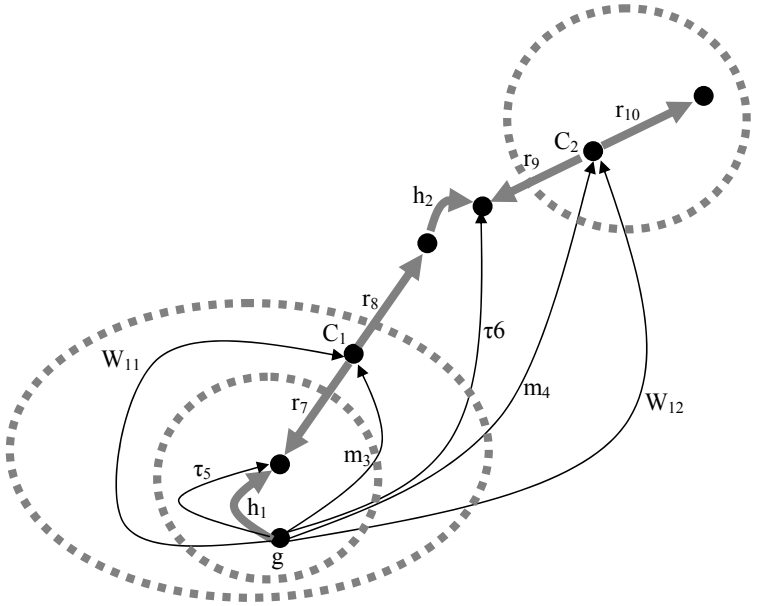


Figure A.5: The Cutsets for the Chord Transformations

$$
\begin{aligned}
&-\vec{F}_7 + \vec{F}_{m3} + \vec{F}_{m4} + \vec{F}_{W12} + \vec{F}_{W11} = 0 \\
&\therefore \vec{F}_7 = \vec{F}_{m3} + \vec{F}_{m4} + \vec{F}_{W12} + \vec{F}_{W11} \\
&\vec{F}_8 + \vec{F}_{m4} + \vec{F}_{W12} = 0 \\
&\therefore \vec{F}_8 = -\vec{F}_{m4} - \vec{F}_{W12} \\
&-\vec{F}_9 + \vec{F}_{m4} + \vec{F}_{W12} = 0 \\
&\therefore \vec{F}_9 = \vec{F}_{m4} + \vec{F}_{W12}
\end{aligned}
$$

The terminal equations must be applied to the f-cutset equations so that each parameter has a physical significance.

$$
\begin{aligned}
\vec{F}_7 &= -m_1 \cdot \vec{a}_1 - m_2 \cdot \vec{a}_2 + \vec{W}_1 + \vec{W}_2 \\
\vec{F}_8 &= m_2 \cdot \vec{a}_2 - \vec{W}_2 \\
\vec{F}_9 &= -m_2 \cdot \vec{a}_2 + \vec{W}_2
\end{aligned}
$$

The secondary variables need to also be eliminated from the equations above. The secondary variables are the two accelerations since the mass elements of the two links are in the cotree. Therefore, the f-circuit equations of both the mass elements are used to create an expression which specifies the accelerations in terms of the primary variables. The f-circuits are shown in Figure (A.6) as the dashed lines. Note the circuits are in the T-tree since the acceleration is a translational component.

The following f-circuit equations are obtained from the two f-circuits.

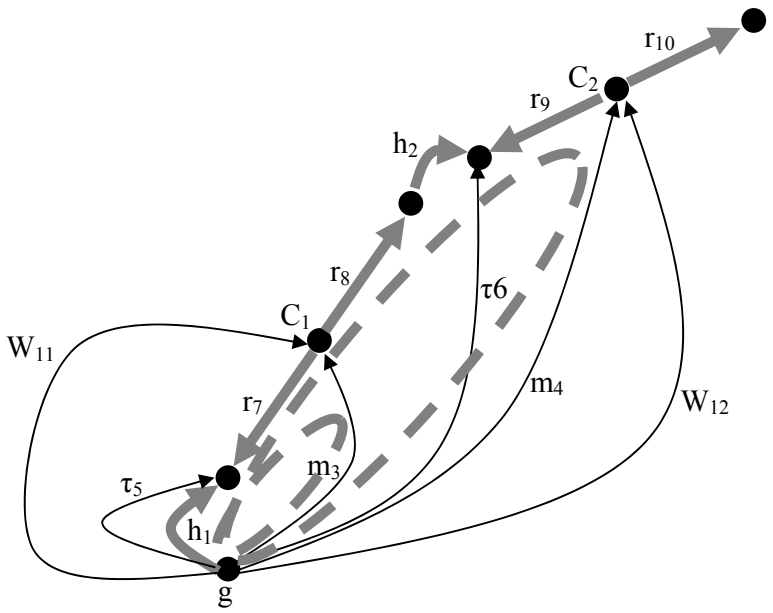


Figure A.6: The Circuits for the Branch Transformations

$$\vec{a}_{m3} + \vec{a}_7 - \vec{a}_{h1} = 0$$
$$\therefore \vec{a}_{m3} = -\vec{a}_7$$
$$\vec{a}_{m4} + \vec{a}_9 - \vec{a}_{h2} - \vec{a}_8 + \vec{a}_7 - \vec{a}_{h1} = 0$$
$$\therefore \vec{a}_{m4} = -\vec{a}_9 + \vec{a}_8 - \vec{a}_7$$

Note the revolute joints do not have a translational component of acceleration; thus the acceleration is automatically zero. Once again the terminal equations need to be substituted into the above equations and doing this leads to the following f-circuit equations.

$$\vec{a}_{m3} = \vec{a}_1 = -\ddot{\vec{q}}_1 \times \vec{r}_7 - \dot{\vec{q}}_1 \times \left(\dot{\vec{q}}_1 \times \vec{r}_7\right)$$
$$\vec{a}_{m4} = \vec{a}_2 = -\left(\ddot{\vec{q}}_1 + \vec{\ddot{q}}_2\right) \times \vec{r}_9 - \left(\dot{\vec{q}}_1 + \dot{\vec{q}}_2\right) \times \left[\left(\dot{\vec{q}}_1 + \dot{\vec{q}}_2\right) \times \vec{r}_9\right]$$
$$+\ddot{\vec{q}}_1 \times \vec{r}_8 + \dot{\vec{q}}_1 \times \left(\dot{\vec{q}}_1 \times \vec{r}_8\right) - \ddot{\vec{q}}_1 \times \vec{r}_7 - \dot{\vec{q}}_1 \times \left(\dot{\vec{q}}_1 \times \vec{r}_7\right)$$

Therefore, the force equations can be updated to include these transformations.

$$\vec{F}_7 = -m_1 \cdot \left(-\ddot{\vec{q}}_1 \times \vec{r}_7 - \dot{\vec{q}}_1 \times \left(\dot{\vec{q}}_1 \times \vec{r}_7\right)\right)$$
$$-m_2 \cdot \begin{pmatrix} -\left(\ddot{\vec{q}}_1 + \ddot{\vec{q}}_2\right) \times \vec{r}_9 - \left(\dot{\vec{q}}_1 + \dot{\vec{q}}_2\right) \times \left[\left(\dot{\vec{q}}_1 + \dot{\vec{q}}_2\right) \times \vec{r}_9\right] + \\ \ddot{\vec{q}}_1 \times \vec{r}_8 + \dot{\vec{q}}_1 \times \left(\dot{\vec{q}}_1 \times \vec{r}_8\right) - \ddot{\vec{q}}_1 \times \vec{r}_7 - \dot{\vec{q}}_1 \times \left(\dot{\vec{q}}_1 \times \vec{r}_7\right) \end{pmatrix} + \vec{W}_1 + \vec{W}_2$$

$$\vec{F}_8 = m_2 \cdot \left(\begin{array}{l} -\left(\ddot{\vec{q}}_1 + \ddot{\vec{q}}_2\right) \times \vec{r}_9 - \left(\dot{\vec{q}}_1 + \dot{\vec{q}}_2\right) \times \left[\left(\dot{\vec{q}}_1 + \dot{\vec{q}}_2\right) \times \vec{r}_9\right] + \ddot{\vec{q}}_1 \times \vec{r}_8 + \\ \dot{\vec{q}}_1 \times \left(\dot{\vec{q}}_1 \times \vec{r}_8\right) - \ddot{\vec{q}}_1 \times \vec{r}_7 - \dot{\vec{q}}_1 \times \left(\dot{\vec{q}}_1 \times \vec{r}_7\right) \end{array} \right) - \vec{W}_2$$

$$\vec{F}_9 = -m_2 \cdot \left(\begin{array}{l} -\left(\ddot{\vec{q}}_1 + \ddot{\vec{q}}_2\right) \times \vec{r}_9 - \left(\dot{\vec{q}}_1 + \dot{\vec{q}}_2\right) \times \left[\left(\dot{\vec{q}}_1 + \dot{\vec{q}}_2\right) \times \vec{r}_9\right] + \\ \ddot{\vec{q}}_1 \times \vec{r}_8 + \dot{\vec{q}}_1 \times \left(\dot{\vec{q}}_1 \times \vec{r}_8\right) - \ddot{\vec{q}}_1 \times \vec{r}_7 - \dot{\vec{q}}_1 \times \left(\dot{\vec{q}}_1 \times \vec{r}_7\right) \end{array} \right) + \vec{W}_2$$

The force equations can then be substituted in the equations of motion so that they are in terms of the primary variables.

$$\left\{ \left(\begin{array}{l} \tau_1 \hat{k}_1 - I_1 \cdot \ddot{\vec{q}}_1 - \dot{\vec{q}}_1 \left(\times I_1 \cdot \dot{\vec{q}}_1\right) + \tau_2 \hat{k}_1 - I_2 \cdot \left(\ddot{\vec{q}}_2 + \ddot{\vec{q}}_1\right) - \left(\dot{\vec{q}}_2 + \dot{\vec{q}}_1\right) \times \left(I_2 \cdot \left(\dot{\vec{q}}_2 + \dot{\vec{q}}_1\right)\right) - \\ \vec{r}_7 \times \left(\begin{array}{l} -m_1 \cdot \left(-\ddot{\vec{q}}_1 \times \vec{r}_7 - \dot{\vec{q}}_1 \times \left(\dot{\vec{q}}_1 \times \vec{r}_7\right)\right) - \\ m_2 \cdot \left(\begin{array}{l} -\left(\ddot{\vec{q}}_1 + \overrightarrow{\ddot{q}}_2\right) \times \vec{r}_9 - \left(\dot{\vec{q}}_1 + \dot{\vec{q}}_2\right) \times \left(\left(\dot{\vec{q}}_1 + \dot{\vec{q}}_2\right) \times \vec{r}_9\right) + \\ \ddot{\vec{q}}_1 \times \vec{r}_8 + \dot{\vec{q}}_1 \times \left(\dot{\vec{q}}_1 \times \vec{r}_8\right) - \ddot{\vec{q}}_1 \times \vec{r}_7 - \dot{\vec{q}}_1 \times \left(\dot{\vec{q}}_1 \times \vec{r}_7\right) \end{array} \right) + \vec{W}_1 + \vec{W}_2 \end{array} \right) - \\ \vec{r}_8 \times \left(m_2 \cdot \left(\begin{array}{l} -\left(\ddot{\vec{q}}_1 + \overrightarrow{\ddot{q}}_2\right) \times \vec{r}_9 - \left(\dot{\vec{q}}_1 + \dot{\vec{q}}_2\right) \times \left(\left(\dot{\vec{q}}_1 + \dot{\vec{q}}_2\right) \times \vec{r}_9\right) + \\ \ddot{\vec{q}}_1 \times \vec{r}_8 + \dot{\vec{q}}_1 \times \left(\dot{\vec{q}}_1 \times \vec{r}_8\right) - \ddot{\vec{q}}_1 \times \vec{r}_7 - \dot{\vec{q}}_1 \times \left(\dot{\vec{q}}_1 \times \vec{r}_7\right) \end{array} \right) - \vec{W}_2 \right) - \\ \vec{r}_9 \times \left(-m_2 \cdot \left(\begin{array}{l} -\left(\ddot{\vec{q}}_1 + \overrightarrow{\ddot{q}}_2\right) \times \vec{r}_9 - \left(\dot{\vec{q}}_1 + \dot{\vec{q}}_2\right) \times \left(\left(\dot{\vec{q}}_1 + \dot{\vec{q}}_2\right) \times \vec{r}_9\right) + \ddot{\vec{q}}_1 \times \vec{r}_8 + \\ \dot{\vec{q}}_1 \times \left(\dot{\vec{q}}_1 \times \vec{r}_8\right) - \ddot{\vec{q}}_1 \times \vec{r}_7 - \dot{\vec{q}}_1 \times \left(\dot{\vec{q}}_1 \times \vec{r}_7\right) \end{array} \right) + \vec{W}_2 \right) \end{array} \right) = 0 \right\} \cdot \hat{k}_1$$

$$\left\{ \left(\begin{array}{l} \tau_2 \hat{k}_2 - I_2 \cdot \left(\ddot{\vec{q}}_2 + \ddot{\vec{q}}_1\right) - \left(\dot{\vec{q}}_2 + \dot{\vec{q}}_1\right) \times \left(I_2 \cdot \left(\dot{\vec{q}}_2 + \dot{\vec{q}}_1\right)\right) - \\ \vec{r}_9 \times \left(-m_2 \cdot \left(\begin{array}{l} -\left(\ddot{\vec{q}}_1 + \overrightarrow{\ddot{q}}_2\right) \times \vec{r}_9 - \left(\dot{\vec{q}}_1 + \dot{\vec{q}}_2\right) \times \left(\left(\dot{\vec{q}}_1 + \dot{\vec{q}}_2\right) \times \vec{r}_9\right) + \\ \ddot{\vec{q}}_1 \times \vec{r}_8 + \dot{\vec{q}}_1 \times \left(\dot{\vec{q}}_1 \times \vec{r}_8\right) - \ddot{\vec{q}}_1 \times \vec{r}_7 - \dot{\vec{q}}_1 \times \left(\dot{\vec{q}}_1 \times \vec{r}_7\right) \end{array} \right) + \vec{W}_2 \right) \end{array} \right) = 0 \right\} \cdot \hat{k}_2$$

Step 13: Transform the equations to a common reference frame (the reference frame in which the motion space has been defined). This is accomplished using the rotation matrices specified in step 6. Then simplify the expressions to obtain the desired equations of motion in terms of the generalized coordinates.

This is the most tedious task of the problem. The second equation needs to be solved first, since the first equation is in terms of both of the torques. The following is the solution for the second torque. Note, the equations must be specified in terms of the frame of the second link since the motion space of the second revolute joint was specified with respect to this frame.

$$\left\{ \left(\begin{array}{l} \tau_2 \hat{k}_2 - I_2 \cdot \left(\ddot{\vec{q}}_2 + \ddot{\vec{q}}_1\right) - \left(\dot{\vec{q}}_2 + \dot{\vec{q}}_1\right) \times \left(I_2 \cdot \left(\dot{\vec{q}}_2 + \dot{\vec{q}}_1\right)\right) - \\ \vec{r}_9 \times \left(-m_2 \cdot \left(\begin{array}{l} -\left(\ddot{\vec{q}}_1 + \ddot{\vec{q}}_2\right) \times \vec{r}_9 - \left(\dot{\vec{q}}_1 + \dot{\vec{q}}_2\right) \times \left(\left(\dot{\vec{q}}_1 + \dot{\vec{q}}_2\right) \times \vec{r}_9\right) + \\ \ddot{\vec{q}}_1 \times \vec{r}_8 + \dot{\vec{q}}_1 \times \left(\dot{\vec{q}}_1 \times \vec{r}_8\right) - \ddot{\vec{q}}_1 \times \vec{r}_7 - \dot{\vec{q}}_1 \times \left(\dot{\vec{q}}_1 \times \vec{r}_7\right) \end{array} \right) + \vec{W}_2 \right) \end{array} \right) = 0 \right\} \cdot \hat{k}_2$$

$$I_2 \cdot \left(\vec{\ddot{q}}_2 + \vec{\ddot{q}}_1\right) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & \frac{1}{12}m_2 l_2^2 & 0 \\ 0 & 0 & \frac{1}{12}m_2 l_2^2 \end{bmatrix} \begin{Bmatrix} 0 \\ 0 \\ \left(\vec{\ddot{q}}_2 + \vec{\ddot{q}}_1\right) \end{Bmatrix} = \tfrac{1}{12}m_2 l_2^2 \left(\vec{\ddot{q}}_2 + \vec{\ddot{q}}_1\right) \hat{k}_2$$

$$\left(\vec{\dot{q}}_2 + \vec{\dot{q}}_1\right) \times \left(I_2 \cdot \left(\vec{\dot{q}}_2 + \vec{\dot{q}}_1\right)\right) = 0$$

$$\vec{r}_9 \times \left(-m_2 \cdot \begin{pmatrix} -\left(\vec{\ddot{q}}_1 + \vec{\ddot{q}}_2\right) \times \vec{r}_9 - \left(\vec{\dot{q}}_1 + \vec{\dot{q}}_2\right) \times \left(\left(\vec{\dot{q}}_1 + \vec{\dot{q}}_2\right) \times \vec{r}_9\right) + \vec{\ddot{q}}_1 \times \vec{r}_8 + \\ \vec{\dot{q}}_1 \times \left(\vec{\dot{q}}_1 \times \vec{r}_8\right) - \vec{\ddot{q}}_1 \times \vec{r}_7 - \vec{\dot{q}}_1 \times \left(\vec{\dot{q}}_1 \times \vec{r}_7\right) \end{pmatrix} + \vec{W}_2 \right)$$

$$= \tfrac{m_2 l_2}{2} \hat{i}_2 \times \begin{Bmatrix} -\left(\ddot{q}_1 + \ddot{q}_2\right) \hat{k}_2 \times \left(\frac{-l_2}{2}\right) \hat{i}_2 - \left(\dot{q}_1 + \dot{q}_2\right) \hat{k}_2 \times \left(\dot{q}_1 + \dot{q}_2\right) \hat{k}_2 \times \left(\frac{-l_2}{2}\right) \hat{i}_2 \\ +\ddot{q}_1 \hat{k}_1 \times \left(\frac{l_1}{2}\right) \hat{i}_1 + \dot{q}_1 \hat{k}_1 \times \left(\dot{q}_1 \hat{k}_1 \times \left(\frac{l_1}{2}\right) \hat{i}_1\right) - \ddot{q}_1 \hat{k}_1 \times \left(\frac{-l_1}{2}\right) \hat{i}_1 - \\ \dot{q}_1 \hat{k}_1 \times \left(\dot{q}_1 \hat{k}_1 \times \left(\frac{-l_1}{2}\right) \hat{i}_1\right) + g\hat{J} \end{Bmatrix}$$

$$= \tfrac{m_2 l_2}{2} \hat{i}_2 \times \left\{ \tfrac{(\ddot{q}_1 + \ddot{q}_2) l_2}{2} \hat{j}_2 - \tfrac{(\dot{q}_1 + \dot{q}_2)^2 l_2}{2} \hat{i}_2 + \ddot{q}_1 l_1 \hat{j}_1 - \dot{q}_1^2 l_1 \hat{i}_1 + g\hat{J} \right\}$$

NOTE, the equations above need to be specified in a common frame

$$= \tfrac{m_2 l_2}{2} \hat{i}_2 \times \begin{Bmatrix} \frac{(\ddot{q}_1 + \ddot{q}_2) l_2}{2} \hat{j}_2 - \frac{(\dot{q}_1 + \dot{q}_2)^2 l_2}{2} \hat{i}_2 + \begin{bmatrix} \cos(q_2) & -\sin(q_2) & 0 \\ \sin(q_2) & \cos(q_2) & 0 \\ 0 & 0 & 1 \end{bmatrix}^T \begin{Bmatrix} -\dot{q}_1^2 l_1 \\ \ddot{q}_1 l_1 \\ 0 \end{Bmatrix} \\ + \begin{bmatrix} \cos(q_1 + q_2) & -\sin(q_1 + q_2) & 0 \\ \sin(q_1 + q_2) & \cos(q_1 + q_2) & 0 \\ 0 & 0 & 1 \end{bmatrix}^T \begin{Bmatrix} 0 \\ g \\ 0 \end{Bmatrix} \end{Bmatrix}$$

$$= \tfrac{m_2 l_2}{2} \hat{i}_2 \times \begin{Bmatrix} -\frac{(\dot{q}_1 + \dot{q}_2)^2 l_2}{2} - \dot{q}_1^2 l_1 \cos(q_2) + \ddot{q}_1 l_1 \sin(q_2) + m_2 g \sin(q_1 + q_2) \\ \frac{(\ddot{q}_1 + \ddot{q}_2) l_2}{2} + \dot{q}_1^2 l_1 \sin(q_2) + \ddot{q}_1 l_1 \cos(q_2) + g \cos(q_1 + q_2) \\ 0 \end{Bmatrix}$$

$$= \left(\tfrac{m_2(\ddot{q}_1 + \ddot{q}_2) l_2^2}{4} + \tfrac{m_2 \dot{q}_1^2 l_1 l_2 \sin(q_2)}{2} + \tfrac{m_2 \ddot{q}_1 l_1 l_2 \cos(q_2)}{2} + \tfrac{m_2 l_2 g \cos(q_1 + q_2)}{2} \right) \cdot \hat{k}_2$$

$\therefore$ combining yields,

$$\left\{ \begin{pmatrix} \tau_2 \hat{k}_2 - \frac{1}{12}m_2 l_2^2 \left(\ddot{q}_2 + \ddot{q}_1\right) \hat{k}_2 - \\ \left(\frac{m_2(\ddot{q}_1 + \ddot{q}_2) l_2^2}{4} + \frac{m_2 \dot{q}_1^2 l_1 l_2 \sin(q_2)}{2} + \frac{m_2 \ddot{q}_1 l_1 l_2 \cos(q_2)}{2} + \frac{m_2 l_2 g \cos(q_1 + q_2)}{2} \right) \cdot \hat{k}_2 \end{pmatrix} = 0 \right\} \cdot \hat{k}_2$$

$$\tau_2 - \tfrac{1}{12}m_2 l_2^2 \left(\ddot{q}_2 + \ddot{q}_1\right) - \begin{pmatrix} \frac{m_2(\ddot{q}_1 + \ddot{q}_2) l_2^2}{4} + \frac{m_2 \dot{q}_1^2 l_1 l_2 \sin(q_2)}{2} + \\ \frac{m_2 \ddot{q}_1 l_1 l_2 \cos(q_2)}{2} + \frac{m_2 l_2 g \cos(q_1 + q_2)}{2} \end{pmatrix} = 0$$

$$\tau_2 = \tfrac{1}{12}m_2 l_2^2 \left(\ddot{q}_2 + \ddot{q}_1\right) + \tfrac{m_2(\ddot{q}_1 + \ddot{q}_2) l_2^2}{4} + \tfrac{m_2 \dot{q}_1^2 l_1 l_2 \sin(q_2)}{2} + \tfrac{m_2 \ddot{q}_1 l_1 l_2 \cos(q_2)}{2} + \tfrac{m_2 l_2 g \cos(q_1 + q_2)}{2}$$

$$\tau_2 = m_2 \left[\tfrac{l_2^2}{3} + \tfrac{l_1 l_2 \cos(q_2)}{2} \right] \ddot{q}_1 + \tfrac{m_2 l_2^2 \ddot{q}_2}{3} + \tfrac{m_2 \dot{q}_1^2 l_1 l_2 \sin(q_2)}{2} + \tfrac{m_2 l_2 g \cos(q_1 + q_2)}{2}$$

The equation for the first torque can be obtained in a similar fashion using the first equation of motion, and after tedious manipulations one would obtain the following.

$$\tau_1 = \left[\tfrac{m_1 l_1^2}{3} + m_2 \left(l_1^2 + \tfrac{l_2^2}{3} + l_1 l_2 \cos(q_2) \right) \right] \ddot{q}_1 + m_2 \left[\tfrac{l_2^2}{3} + \tfrac{l_1 l_2 \cos(q_2)}{2} \right] \ddot{q}_2$$
$$- \tfrac{m_2 l_1 l_2 \sin(q_2)}{2} \dot{q}_2^2 - m_2 l_1 l_2 \sin(q_2) \dot{q}_1 \dot{q}_2 + \left(\tfrac{m_1}{2} + m_2 \right) g l_1 \cos(q_1) + \tfrac{m_2}{2} g l_2 \cos(q_1 + q_2)$$

A.2 Solution Using Newton-Euler Equations

Steps 1 – 7: Same as the solution using linear graph theory.

Step 8: Draw a free body diagram (FBD) for each of the bodies in the system with respect to the inertial frame of each body.

For the two-link planar manipulator, the two free body diagrams are shown in Figure (A.7). It is important to note that the FBDs were drawn in the XY plane, and that there is a reaction in the z direction on both revolute joints which should be added to the FBDs.

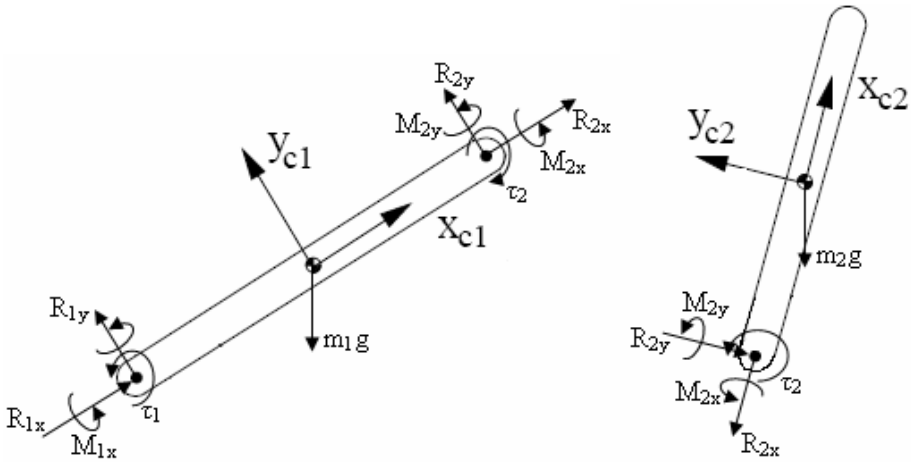


Figure A.7: Free Body Diagrams of the Planar Manipulator

Step 9: Apply Newton-Euler equations to each body in the direction of the generalized coordinates to get the equations of motion.

For the two link manipulator, the direction of the generalized coordinates is a rotation about the Z axis, thus Newton-Euler equations will be applied to each link in this direction. It is to be noted that the translational equations do not need to be considered since the motion of each generalized coordinate is a rotational motion.

$$\left(\sum \vec{M}_1 = \mathbf{I_1} \cdot \ddot{\vec{q}}_1 + \dot{\vec{q}}_1 \times \left(\mathbf{I_1} \cdot \dot{\vec{q}}_1\right)\right) \cdot \hat{K}$$
$$\left(R_{2y}\tfrac{l_1}{2} - R_{1y}\tfrac{l_1}{2} + \tau_1 - \tau_2\right) = \tfrac{1}{12} m_1 l_1^2 \ddot{q}_1$$
$$\left(\sum \vec{M}_2 = \mathbf{I_2} \cdot \left(\ddot{\vec{q}}_1 + \ddot{\vec{q}}_2\right) + \left(\dot{\vec{q}}_1 + \dot{\vec{q}}_2\right) \times \left(\mathbf{I_2} \cdot \left(\dot{\vec{q}}_1 + \dot{\vec{q}}_2\right)\right)\right) \cdot \hat{K}$$
$$\left(R_{2y}\tfrac{l_2}{2} + \tau_2\right) = \tfrac{1}{12} m_2 l_2^2 \left(\ddot{q}_1 + \ddot{q}_2\right)$$

Note, the dot product was taken right away, and some of the computations have already been computed when solving the problem using LGT.

Step 10: Eliminate the constraint forces from the equations of motion. This is accomplished by applying Newton-Euler equations in the appropriate direction of

the constraint force; e.g. if the constraint force is applied to body 1 and along the x axis than apply Newton-Euler equations in this direction to eliminate the constraint force. It is also important to note that the reaction forces need to be in terms of the generalized coordinates, thus use kinematics when appropriate in order to accomplish this. Note it is also important that the equations are expressed in a common frame, and the rotation matrices can be used to accomplish this.

$$\begin{array}{l}
\left(R_{2y}\frac{l_2}{2}+\tau_2\right)=\frac{1}{12}m_2l_2^2\left(\ddot{q}_1+\ddot{q}_2\right)\\
\tau_2=\frac{1}{12}m_2l_2^2\left(\ddot{q}_1+\ddot{q}_2\right)-R_{2y}\frac{l_2}{2}\\
\sum\vec{F}_2=m_2\vec{a}_2\\
m_2\vec{a}_2=-R_{2y}\hat{j}_2-R_{2x}\hat{i}_2-m_2g\hat{J}
\end{array}$$

Note the rotation matrix is needed to transfer the weight from components in the global frame to components in the frame of the second body. Also, note that kinematic equations are needed in order to express the acceleration of the second link in terms of the generalized coordinates.

$$\vec{W}_2=-m_2g\hat{J}=\begin{bmatrix}\cos\left(q_1+q_2\right)&-\sin\left(q_1+q_2\right)&0\\\sin\left(q_1+q_2\right)&\cos\left(q_1+q_2\right)&0\\0&0&1\end{bmatrix}^T\begin{Bmatrix}0\\-m_2g\\0\end{Bmatrix}=$$

$$\begin{Bmatrix}-m_2g\sin\left(q_1+q_2\right)\\-m_2g\cos\left(q_1+q_2\right)\\0\end{Bmatrix}$$

$$\vec{W}_2=-m_2g\sin\left(q_1+q_2\right)\hat{i}_2-m_2g\cos\left(q_1+q_2\right)\hat{j}_2$$

$$m_2\vec{a}_2=-R_{2y}\hat{j}_2-R_{2x}\hat{i}_2-m_2g\sin\left(q_1+q_2\right)\hat{i}_2-$$

$$m_2g\cos\left(q_1+q_2\right)\hat{j}_2=m_2\left(a_{2x}\hat{i}_2+a_{2y}\hat{j}_2+a_{2x}\hat{k}_2\right)$$

$$m_2\left(a_{2x}\hat{i}_2+a_{2y}\hat{j}_2+a_{2x}\hat{k}_2\right)=\begin{pmatrix}-R_{2y}\hat{j}_2-R_{2x}\hat{i}_2-\\m_2g\sin\left(q_1+q_2\right)\hat{i}_2-m_2g\cos\left(q_1+q_2\right)\hat{j}_2\end{pmatrix}\cdot\hat{j}_2$$

$$\hat{j}_2\Rightarrow m_2a_{2y}=-R_{2y}-m_2g\cos\left(q_1+q_2\right)$$

$$\vec{a}_2=\vec{a}_{Rev,2}+\vec{a}_{2/Rev,2}$$

$$\vec{a}_{Rev,2}=\vec{a}_{Rev,1}+\vec{a}_{Rev,2/Rev,1}=\vec{a}_{Rev,2/Rev,1}=$$

$$\ddot{q}_1\hat{k}_1\times l_1\hat{i}_1+\dot{q}_1\hat{k}_1\times\left(\dot{q}_1\hat{k}_1\times l_1\hat{i}_1\right)=\ddot{q}_1l_1\hat{j}_1-\dot{q}_1^2l_1\hat{i}_1$$

$$\vec{a}_{Rev,2} = \begin{bmatrix} \cos(q_2) & -\sin(q_2) & 0 \\ \sin(q_2) & \cos(q_2) & 0 \\ 0 & 0 & 1 \end{bmatrix}^T \begin{Bmatrix} -\dot{q}_1^2 l_1 \\ \ddot{q}_1 l_1 \\ 0 \end{Bmatrix} = \begin{Bmatrix} -\dot{q}_1^2 l_1 \cos(q_2) + \ddot{q}_1 l_1 \sin(q_2) \\ \dot{q}_1^2 l_1 \sin(q_2) + \ddot{q}_1 l_1 \cos(q_2) \\ 0 \end{Bmatrix}$$

$$\vec{a}_{Rev,2} = \left(-\dot{q}_1^2 l_1 \cos(q_2) + \ddot{q}_1 l_1 \sin(q_2)\right)\hat{i}_2 + \left(\dot{q}_1^2 l_1 \sin(q_2) + \ddot{q}_1 l_1 \cos(q_2)\right)\hat{j}_2$$

$$\vec{a}_{2/Rev,2} = \left(\ddot{q}_1 + \ddot{q}_2\right)\hat{k}_2 \times \tfrac{l_2}{2}\hat{i}_2 + \left(\dot{q}_1 + \dot{q}_2\right)\hat{k}_2 \times \left(\left(\dot{q}_1 + \dot{q}_2\right)\hat{k}_2 \times \tfrac{l_2}{2}\hat{i}_2\right)$$

$$\vec{a}_{2/Rev,2} = \tfrac{(\ddot{q}_1+\ddot{q}_2)l_2}{2}\hat{j}_2 - \tfrac{(\dot{q}_1+\dot{q}_2)^2 l_2}{2}\hat{i}_2$$

$$\vec{a}_2 = \vec{a}_{Rev,2} + \vec{a}_{2/Rev,2}$$

$$\vec{a}_2 = \begin{pmatrix} -\dot{q}_1^2 l_1 \cos(q_2) + \\ \ddot{q}_1 l_1 \sin(q_2) \end{pmatrix}\hat{i}_2 + \begin{pmatrix} \dot{q}_1^2 l_1 \sin(q_2) + \\ \ddot{q}_1 l_1 \cos(q_2) \end{pmatrix}\hat{j}_2 + \tfrac{(\ddot{q}_1+\ddot{q}_2)l_2}{2}\hat{j}_2 - \tfrac{(\dot{q}_1+\dot{q}_2)^2 l_2}{2}\hat{i}_2$$

$$a_{2y} = \vec{a}_2 \cdot \hat{j}_2 = \dot{q}_1^2 l_1 \sin(q_2) + \ddot{q}_1 l_1 \cos(q_2) + \tfrac{(\ddot{q}_1+\ddot{q}_2)l_2}{2}$$

$$\therefore R_{2y} = -m_2\dot{q}_1^2 l_1 \sin(q_2) - m_2\ddot{q}_1 l_1 \cos(q_2) - \tfrac{m_2(\ddot{q}_1+\ddot{q}_2)l_2}{2} - m_2 g\cos(q_1+q_2)$$

$$\therefore \tau_2 = \tfrac{1}{12}m_2 l_2^2\left(\ddot{q}_1 + \ddot{q}_2\right) + \tfrac{l_2}{2}\begin{pmatrix} m_2\dot{q}_1^2 l_1 \sin(q_2) + m_2\ddot{q}_1 l_1 \cos(q_2) + \\ \tfrac{m_2(\ddot{q}_1+\ddot{q}_2)l_2}{2} + m_2 g\cos(q_1+q_2) \end{pmatrix}$$

$$\tau_2 = m_2\left[\tfrac{l_2^2}{3} + \tfrac{l_1 l_2\cos(q_2)}{2}\right]\ddot{q}_1 + \tfrac{m_2 l_2^2}{3}\ddot{q}_2 + \tfrac{m_2 l_1 l_2\sin(q_2)}{2}\dot{q}_1^2 + \tfrac{m_2 l_2 g\cos(q_1+q_2)}{2}$$

Note, that the expression for the second torque is the same as that obtained from LGT. Also in a similar fashion the first joint torque can be found to be the same as that specified above for LGT solutions.

It can be noted that there are more steps to obtained the solution using linear graph theory, but it is a more systematical approach. Therefore, a computer implementation can be created based on the approach to solve mutlibody systems [65].

Appendix B

Tire Model Parameters

The parameters used in the Fiala and Pacejeka tire models throughout this thesis are presented in this section along with the parameters used in the stretched string equations (Equations (3.26) and (3.27)) when tire transients are important. The parameters used to estimate the effective rolling radius using Equation (3.16) are also presented in this section.

Parameter	Value	Description
F_z0	5900	Nominal normal force (N)
B	8.4	Low load stiffness parameter
D	0.27	Peak value of R_{eff}
F	0.07	High load stiffness parameter
k_Z	21000	Tire stiffness $(\frac{N}{m})$
R_u	0.344	Unloaded radius (m)

Table B.1: Parameters for Calculating the Effective Rolling Radius

Parameter	Value	Description
D_2	0.16	Width of tire (m)
C_S	115000	Longitudinal stiffness (N)
C_α	117000	Lateral stiffness (N/rad)
C_r	0.01	Coefficient of rolling resistance (m)
μ_0	1.22	Peak coefficient of friction
μ_1	0.2	Sliding coefficient of friction

Table B.2: Fiala Tire Model Parameters

Parameter	Value	Description
F_{z0}	5900	Nominal normal force
R_0	0.355	Unloaded radius
V_0	22.2	Reference Velocity
LF_{z0}	1	Scale factor of nominal load
LC_x	1	Scale factor of F_x shape factor
$L_{\mu x}$	1	Scale factor F_x of peak friction coefficient
LE_x	1	Scale factor of F_x curvature factor
LK_x	1	Scale factor of F_x slip stiffness
LH_x	1	Scale factor of F_x horizontal shift
LV_x	1	Scale factor of F_x vertical shift
$L_{\gamma x}$	1	Scale factor of camber for F_x
LC_y	1	Scale factor of M_z shape factor
$L\mu_y$	1	Scale factor of M_z peak friction coefficient
LE_y	1	Scale factor of curvature factor
LK_y	1	Scale factor of M_z cornering stiffness
LH_y	1	Scale factor of M_z horizontal shift
LV_y	1	Scale factor of M_z vertical shift
$L\gamma_y$	1	Scale factor of camber for M_z
L_{trail}	1	Scale factor of peak pneumatic trail
L_{res}	1	Scale factor for offset of residual torque
$L_{\gamma z}$	1	Scale factor of camber for M_z
LX_α	1	Scale factor of slip angle influence on F_x
LY_κ	1	Scale factor of longitudinal slip influence on M_z
$LV_{y\kappa}$	1	Scale factor of longitudinal slip influence on M_z
LS	1	Scale factor of moment arm of F_x about vertical axis
LM_x	1	Scale factor of overturning couple
LVM_x	1	Scale factor of M_x vertical shift
LM_y	1	Scale factor of rolling resistance torque
PC_{x1}	1.65	Shape factor for longitudinal force
PD_{x1}	1	Longitudinal friction, μ_x, at F_{z0} and zero inclination
PD_{x2}	0	Variation of friction, μ_x, with load
PD_{x3}	0	Variation of friction, μ_x, with inclination
PE_{x1}	0	Longitudinal curvature EF_x at F_{z0}
PE_{x2}	0	Variation of curvature EF_x with load
PE_{x3}	0	Variation of curvature EF_x with load squared
PE_{x4}	0	Brake/drive asymmetry factor for EF_z
PK_{x1}	20	Longitudinal slip stiffness $\frac{Kf_x}{F_z}$ at F_{z0}
PK_{x2}	0	Variation of slip stiffness $\frac{Kf_x}{F_z}$ with load
PK_{x3}	0	Exponent in slip stiffness $\frac{Kf_x}{F_z}$ with load

*Note, the units for the parameters in this table follow the N, m, s, and rad system

Table B.3: Pacejka Tire Model Parameters

Parameter	Value	Description
PH_{x1}	0	Horizontal shift of longitudinal slip at F_{z0}
PH_{x2}	0	Variation of horizontal shift with load
PV_{x1}	0	Vertical shift at F_{z0}
PV_{x2}	0	Variation of vertical shift with load
RB_{x1}	1	Slope factor for combined slip F_x reduction
RB_{x2}	6	Variation of slope F_x reduction with longitudinal slip
RC_{x1}	1	Shape factor for combined slip F_x reduction
RE_{x1}	0	Curvature factor for combined slip F_x reduction
RE_{x2}	0	Variation of curvature factor with load
RH_{x1}	0	Shift factor for combined slip F_x reduction
QS_{x1}	0	Vertical force induced overturning moment
QS_{x2}	0	Camber induced overturning couple
QS_{x3}	0	Aligning moment induced overturning couple
PC_{y1}	1.1038	Shape factor for pure lateral force
PD_{y1}	1.0491	Lateral peak friction, μ_y
PD_{y2}	-0.2153	Variation of μ_y with load
PD_{y3}	-0.3885	Variation of μ_y with inclination squared
PE_{y1}	-0.8715	Lateral force curvature factor at F_{z0}
PE_{y2}	-1.5278	Variation of curvature with load
PE_{y3}	-0.0260	Dependency of curvature on the sign of slip angle
PE_{y4}	-0.9073	Variation of curvature with camber
PK_{y1}	-23.648	Maximum value of cornering stiffness $\frac{KM_z}{F_{znom}}$
PK_{y3}	2.1393	Load for max cornering stiffness
PK_{y3}	-0.9073	Variation of KM_z with camber
PH_{y1}	-1.7802E-4	Horizontal shift of lateral force at F_{z0}
PH_{y2}	-3.4980E-5	Variation of horizontal shift with load
PH_{y3}	2.906E-2	Variation of horizontal shift with camber
PV_{y1}	-4.5310E-4	Vertical shift $\frac{S_{vy}}{F_z}$ at F_{z0}
PV_{y2}	5.4083E-4	Variation of vertical shift with load
PV_{y3}	-0.2062	Variation of vertical shift with camber
PV_{y4}	-0.1557	Variation of vertical shift with camber and load
RB_{y1}	16	Slope factor for combined slip M_z reduction
RB_{y2}	0	Variation of slope factor with slip angle
RB_{y3}	0	Shift term for alpha in slope factor
RC_{y1}	1	Shape factor for combined slip M_z reduction
RE_{y1}	0	Curvature factor for combined slip M_z reduction
RE_{y2}	0	Variation of curvature factor with load
RH_{y1}	0	Shift factor for combined slip M_z reduction
RH_{y2}	0	Variation of shift factor with load

*Note, the units for the parameters in this table follow the N, m, s, and rad system

Table B.4: Pacejka Tire Model Parameters (Continued)

Parameter	Value	Description
RV_{y1}	0	Longitudinal slip induced side forces at F_{z0}
RV_{y2}	0	Variation of vertical shift with load
RV_{y3}	0	Variation of vertical shift with camber
RV_{y4}	0	Variation of vertical shift with slip angle
RV_{y5}	1.9	Variation of vertical shift with longitudinal slip
RV_{y6}	0	Variation of vertical shift with arctan (S)
QS_{y1}	0.01	Rolling resistance of torque coefficient
QS_{y2}	0	Rolling resistance influenced by F_x
QS_{y3}	0	Rolling resistance influenced by V_{Cx}
QS_{y4}	0	Rolling resistance influenced by V_{Cx}
QB_{z1}	14.973	Pneumatic trail slope factor, B_t, at F_{z0}
QB_{z2}	-3.5918	Variation of B_t with load
QB_{z3}	-0.7270	Variation of B_t with load squared
QB_{z4}	-7.7765E-3	Variation of B_t with camber
QB_{z5}	-8.1978E-3	Variation of B_t with absolute camber
QB_{z9}	17.781	Slope factor for residual torque B_r
QB_{z10}	0	Slope factor for residual torque B_r
QC_{z1}	1.1320	Shape factor for pneumatic trail
QD_{z1}	0.1127	Controls Peak pneumatic trail, D_t
QD_{z2}	-2.6745E-3	Variation of peak D_t with load
QD_{z3}	-6.6332E-3	Variation of peak D_t with camber
QD_{z4}	-6.2380	Variation of peak D_t with camber squared
QD_{z6}	3.1508E-4	Controls peak residual torque, D_r
QD_{z7}	2.9610E-4	Variation of peak D_r with load
QD_{z8}	-8.1607E-2	Variation of peak R_r with camber
QD_{z9}	-8.0160E-2	Variation peak D_r with camber and load
QE_{z1}	-1.7447	Pneumatic trail curvature at F_{z0}
QE_{z2}	-0.5853	Variation of trail curvature with load
QE_{z3}	0	Variation of trail curvature with load squared
QE_{z4}	2.9202E-3	Variation of trail curvature with sign of α
QE_{z5}	-2.2229	Variation of trail curvature with γ and sign of α
QH_{z1}	2.5222E-4	Pneumatic trail horizontal shift at F_{z0}
QH_{z2}	6.8468E-5	Variation of trail horizontal shift with load
QH_{z3}	-1.9273E-2	Variation of trail horizontal shift with γ
QH_{z4}	9.2761E-3	Variation of trail horizontal shift with γ and load
SS_{z1}	0	Nominal value of $\frac{S}{R_0}$: effect of F_x on M_z
SS_{z2}	0	Variation of S with M_z
SS_{z3}	0	Variation of S with camber
SS_{z4}	0	Variation of S with load and camber

*Note, the units for the parameters in this table follow the N, m, s, and rad system

Table B.5: Pacejka Tire Model Parameters (Continued)

Parameter	Value	Description
F_z0	5900	Nominal normal force (N)
R_0	0.355	Unloaded radius (m)
LS_k	1	Scale factor for B_{long}
LS_α	1	Scale factor for B_{lat}
PT_{x1}	2.3657	B_{long} at nominal load
PT_{x2}	1.4112	Variation of B_{long} with load
PT_{x3}	0.5663	Variation of B_{long} with exponent load
PT_{y1}	2.1439	Peak value of B_{lat}
PT_{y2}	1.9829	Value of F_z/F_{z0} at which B_{lat} is extreme
PK_{y3}	-0.9073	Variation of B_{lat} with inclination

Table B.6: Parameters for Calculating the Relaxation Lengths When the Stretched String Equations are Used

Appendix C

Model Parameters and Initial Conditions

The parameters used to model the Chevrolet Equinox are presented in this section along with the set of initial conditions that were used to solve for the response of the vehicle system. Note, all body-fixed reference frames were initially aligned with the Ground (inertial) reference frame. The inertial reference is shown in Figure (2.2). The section also includes the parameters that were used in the analysis performed on the moving quarter car model.

Parameter	Value	Description
m_s	519.25	Mass of the sprung mass (kg)
m_u	25	Mass of the unsprung mass (kg)
L_s	0.6641	Undeformed length of the suspension spring (m)
k_s	4.8299E4	Stiffness of the suspension spring $(\frac{N}{m})$
k_t	3.04E5	Stiffness of the tire $(\frac{N}{m})$
C_s	3075	Damping of the suspension spring $(\frac{Ns}{m})$
R_u	0.355	Undeformed radius of the tire (m)
δ_i	1.7903E-3	Initial deflection of the tire (m)

Table C.1: Model Parameters of the Moving Quarter Car Model

State Variable	Initia Condition	Description
$v_s(t)$	0	Z component of the velocity of the sprung mass[1] $(\frac{m}{s})$
$v_u(t)$	0	Z component of the velocity of the unsprung mass[1] $(\frac{m}{s})$
$v_t(t)$	2	X component of the velocity of the tire[2] $(\frac{m}{s})$
$Z_s(t)$	0	Z position of the sprung mass[1] (m)
$Z_u(t)$	0	Z position of the unsprung mass[1] (m)
$X_t(t)$	0	X position of the tire[2] (m)

1: With respect to the body fixed frame
2: With respect to the inertial reference frame

Table C.2: Initial Conditions of the Moving Quarter Car Model

Body	Mass (kg)	Rotational Inertial (kgm^2)					
		I_{xx}	I_{yy}	I_{zz}	I_{xy}	I_{xz}	I_{yz}
Vehicle body	2077	330	1925	1925	0	110	0
Lumped mass[1]	10	1	0.5	1	0	0	0
Front knuckle	0	0	0	0	0	0	0
Tires[1]	28	0.78	1.56	0.78	0	0	0

1: Applies to all components

Table C.3: Inertia Properties of the Chevrolet Equinox

Rigid Arm	Translation (m)			Rotation[1] (rad)		
	x	y	z	θ_1	θ_2	θ_3
Chassis COM to FL[2] corner	1.353	0.760	0	0	0	0
Chassis COM to FR[2] corner	1.353	-0.760	0	0	0	0
Chassis COM to RL[2] corner	-1.487	0.795	0	0	0	0
Chassis COM to RR[2] corner	-1.487	-0.795	0	0	0	0
Ground to Road Frame Q	0	0	0	0	0	0

1. These are the Euler angles used to track the orientation between two frames
2. Abbreviation example: FL = Front Left

Table C.4: Rigid Arm Properties of the Chevrolet Equinox

Spring/Damper	Stiffness $(\frac{N}{m})$	Free Length (m)	Damping $(\frac{Ns}{m})$
Front suspension[1]	48299	0.674	3075
Rear suspension[1]	30518	0.720	2331
Tire[2]	3.04E5	0.355	500

1: Applies to both sides of the vehicle
2: Applies to all components

Table C.5: Spring/Damping Properties of the Chevrolet Equinox

State Variable	Initial Condition	Description
$v_x(t)$	20	X component of the velocity of the chassis[1]
$v_y(t)$	0	Y component of the velocity of the chassis[1]
$v_z(t)$	0	Z component of the velocity of the chassis[1]
$\omega_1(t)$	0	X component of the angular velocity of the chassis[1]
$\omega_2(t)$	0	Y component of the angular velocity of the chassis[1]
$\omega_3(t)$	0	Z component of the angular velocity of the chassis[1]
$\frac{d}{dt}s_1(t)$	0	Rate of extension of the FL[5] suspension[4]
$\frac{d}{dt}s_2(t)$	0	Rate of extension of the FR[5] suspension[4]
$\frac{d}{dt}s_3(t)$	0	Rate of extension of the RL[5] suspension[4]
$\frac{d}{dt}s_4(t)$	0	Rate of extension of the RR[5] suspension[4]
$\frac{d}{dt}phi_1(t)$	57.455	Spin rate of the FL[5] tire[4]
$\frac{d}{dt}phi_2(t)$	57.455	Spin rate of the FR[5] tire[4]
$\frac{d}{dt}phi_3(t)$	57.455	Spin rate of the RL[5] tire[4]
$\frac{d}{dt}phi_4(t)$	57.455	Spin rate of the RR[5] tire[4]
$\frac{d}{dt}s_{FL}(t)$	0	The rate of change of the distance traveled along the path of the road by the FL[5] tire[3]
$\frac{d}{dt}s_{FR}(t)$	0	The rate of change of the distance traveled along the path of the road by the FR[5] tire[3]
$\frac{d}{dt}s_{RL}(t)$	0	The rate of change of the distance traveled along the path of the road by the RL[5] tire[3]
$\frac{d}{dt}s_{RR}(t)$	0	The rate of change of the distance traveled along the path of the road by the RR[5] tire[3]

1: With respect to the body fixed frame
2: With respect to the inertial reference frame
3. Only present when the road is defined in tabular form by specifying the location of the centerline of the road
4. Along the joint axis
5. Abbreviation example: FL = Front Left

Note, the table continues on the following page.

Table C.6: Initial Conditions of the Chevrolet Equinox

State Variable	Initial Condition	Description
$x(t)$	0	X position of the chassis[2]
$y(t)$	0	Y position of the chassis[2]
$z(t)$	0.8995	Z position of the chassis[2]
$yaw(t)$	0	First 321 Euler angle of the chassis
$pitch(t)$	0	Second 321 Euler angle of the chassis
$roll(t)$	0	Third 321 Euler angle of the chassis
$s_1(t)$	0.5632	Initial extension of the FL[5] suspension[4]
$s_2(t)$	0.5632	Initial extension of the FR[5] suspension [4]
$s_3(t)$	0.5617	Initial extension of the RL[5] suspension [4]
$s_4(t)$	0.5617	Initial extension of the RR[5] suspension [4]
$phi_1(t)$	0	Rotation angle of the FL[5] tire [4]
$phi_2(t)$	0	Rotation angle of the FR[5] tire [4]
$phi_3(t)$	0	Rotation angle of the RL[5] tire [4]
$phi_4(t)$	0	Rotation angle of the RR[5] tire [4]
$s_{FL}(t)$	0	The distance traveled along the path of the road by the FL[5] tire[3]
$s_{FR}(t)$	0	The distance traveled along the path of the road by the FR[5] tire[3]
$s_{RL}(t)$	0	The distance traveled along the path of the road by the RL[5] tire[3]
$s_{RR}(t)$	0	The distance traveled along the path of the road by the RR[5] tire[3]

1: With respect to the body fixed frame
2: With respect to the inertial reference frame
3. Only present when the road is defined in tabular form by specifying the location of the centerline of the road
4. Along the joint axis
5. Abbreviation example: FL = Front Left

Table C.7: Initial Conditions of the Chevrolet Equinox (Continued)

Appendix D

Road Data Files

The road data files that were used to represent the road in some of the examples is presented in this section. Note, all data points are with respect to the inertial reference frame.

x	z	zp_x^1	α
0	0	0	0
80	0	0	0
90	1	0	0
100	2	0	0
110	3	0	0
120	4	0	0
130	5	0	0
140	6	0	0
150	7	0	0
160	8	0	0
170	9	0	0
180	10	0	0
190	11	0	0
200	12	0	0
300	22	0	0

1. The second derivative of z with respect to x

Table D.1: The Road Data File that was Used in Example 4

The road data file used in MSC.ADAMS to represent the road in Example 4 is shown below. The start and finish of the data file is noted.

Start of MSC.ADAMS road data file

$ –MDI_HEADER
[MDI_HEADER]
FILE_TYPE = 'rdf'

```
FILE_VERSION = 5.00
FILE_FORMAT = 'ASCII'
(COMMENTS)
comment_string)
'3d smooth road - ramp example'
$ – – – – – – – – – – – – – – – – – – – – – – –UNITS
[UNITS]
LENGTH = 'meter'
FORCE = 'newton'
ANGLE = 'radians'
MASS = 'kg'
TIME = 'sec'
$ – – – – – – – – – – – – – – – – – – – – – – –DEFINITION
[MODEL]
METHOD = '3D_SPLINE'
FUNCTION_NAME = 'ARC903'
VERSION = 1.00
$ – – – – – – – – – – – – – – – – – – – – – – –ROAD_PARAMETERS
[GLOBAL_PARAMETERS]
CLOSED_ROAD = 'no'
SEARCH_ALGORITHM = 'SLOW'
ROAD_VERTICAL = '0.0 0.0 1.0'
FORWARD_DIR = 'NORMAL'
$ – – – – – – – – – – – – – – – – – – – – – – –DATA_POINTS
[DATA_POINTS]
X Y Z WIDTH BANK MU_LEFT MU_RIGHT
0.000 0.000 0.000 12.000 0.000 1.000 1.000
5.000 0.000 0.000 12.000 0.000 1.000 1.000
10.000 0.000 0.000 12.000 0.000 1.000 1.000
15.000 0.000 0.000 12.000 0.000 1.000 1.000
20.000 0.000 0.000 12.000 0.000 1.000 1.000
25.000 0.000 0.000 12.000 0.000 1.000 1.000
30.000 0.000 0.000 12.000 0.000 1.000 1.000
35.000 0.000 0.000 12.000 0.000 1.000 1.000
40.000 0.000 0.000 12.000 0.000 1.000 1.000
45.000 0.000 0.000 12.000 0.000 1.000 1.000
50.000 0.000 0.000 12.000 0.000 1.000 1.000
55.000 0.000 0.000 12.000 0.000 1.000 1.000
60.000 0.000 0.000 12.000 0.000 1.000 1.000
65.000 0.000 0.000 12.000 0.000 1.000 1.000
70.000 0.000 0.000 12.000 0.000 1.000 1.000
75.000 0.000 0.000 12.000 0.000 1.000 1.000
80.000 0.000 0.000 12.000 0.000 1.000 1.000 RAMP
85.000 0.000 0.000 12.000 0.000 1.000 1.000
90.000 0.000 0.000 12.000 0.000 1.000 1.000
```

95.000 0.000 0.000 12.000 0.000 1.000 1.000
100.000 0.000 0.000 12.000 0.000 1.000 1.000
105.000 0.000 0.000 12.000 0.000 1.000 1.000
110.000 0.000 0.000 12.000 0.000 1.000 1.000
115.000 0.000 0.000 12.000 0.000 1.000 1.000
120.000 0.000 0.000 12.000 0.000 1.000 1.000
125.000 0.000 0.000 12.000 0.000 1.000 1.000
130.000 0.000 0.000 12.000 0.000 1.000 1.000
135.000 0.000 0.000 12.000 0.000 1.000 1.000
140.000 0.000 0.000 12.000 0.000 1.000 1.000
145.000 0.000 0.000 12.000 0.000 1.000 1.000
150.000 0.000 0.000 12.000 0.000 1.000 1.000
155.000 0.000 0.000 12.000 0.000 1.000 1.000
160.000 0.000 0.000 12.000 0.000 1.000 1.000
165.000 0.000 0.000 12.000 0.000 1.000 1.000
170.000 0.000 0.000 12.000 0.000 1.000 1.000
175.000 0.000 0.000 12.000 0.000 1.000 1.000
180.000 0.000 0.000 12.000 0.000 1.000 1.000
185.000 0.000 0.000 12.000 0.000 1.000 1.000
190.000 0.000 0.000 12.000 0.000 1.000 1.000
195.000 0.000 0.000 12.000 0.000 1.000 1.000
200.000 0.000 0.000 12.000 0.000 1.000 1.000
205.000 0.000 0.000 12.000 0.000 1.000 1.000
210.000 0.000 0.000 12.000 0.000 1.000 1.000
215.000 0.000 0.000 12.000 0.000 1.000 1.000
220.000 0.000 0.000 12.000 0.000 1.000 1.000
225.000 0.000 0.000 12.000 0.000 1.000 1.000
230.000 0.000 0.000 12.000 0.000 1.000 1.000
235.000 0.000 0.000 12.000 0.000 1.000 1.000
240.000 0.000 0.000 12.000 0.000 1.000 1.000
245.000 0.000 0.000 12.000 0.000 1.000 1.000
250.000 0.000 0.000 12.000 0.000 1.000 1.000
255.000 0.000 0.000 12.000 0.000 1.000 1.000
260.000 0.000 0.000 12.000 0.000 1.000 1.000
265.000 0.000 0.000 12.000 0.000 1.000 1.000
270.000 0.000 0.000 12.000 0.000 1.000 1.000
275.000 0.000 0.000 12.000 0.000 1.000 1.000
280.000 0.000 0.000 12.000 0.000 1.000 1.000
285.000 0.000 0.000 12.000 0.000 1.000 1.000
290.000 0.000 0.000 12.000 0.000 1.000 1.000
295.000 0.000 0.000 12.000 0.000 1.000 1.000
300.000 0.000 0.000 12.000 0.000 1.000 1.000
$ −−END_DATA_POINTS
[RAMP]
COORDINATE_SYSTEM = 'local'

START = '80 0 0'
STOP = '500 0 0'
LENGTH = 420
WIDTH = 12
FRICTION = 1.0
ROAD_TYPE = 'RAMP'
HEIGHT = 42
SLOPE = 0.1

End of MSC.ADAMS road data file

x	y	z	zp_s^1	α
0	0.0000	0	0	0
1	0.0050	0	0	0
2	0.0200	0	0	0
3	0.0450	0	0	0
4	0.0800	0	0	0
5	0.1251	0	0	0
6	0.1802	0	0	0
7	0.2453	0	0	0
8	0.3205	0	0	0
9	0.4058	0	0	0
10	0.5013	0	0	0
11	0.6068	0	0	0
12	0.7226	0	0	0
13	0.8486	0	0	0
14	0.9848	0	0	0
15	1.1314	0	0	0
16	1.2883	0	0	0
17	1.4556	0	0	0
18	1.6333	0	0	0
19	1.8216	0	0	0
20	2.0204	0	0	0
21	2.2299	0	0	0
22	2.4500	0	0	0
23	2.6809	0	0	0
24	2.9227	0	0	0
25	3.1754	0	0	0

1. The second derivative of z with respect to s

Table D.2: The Road Data File that was Used in Example 5

x	y	z	zp_s^1	α
26	3.4391	0	0	0
27	3.7140	0	0	0
28	4.0000	0	0	0
29	4.2973	0	0	0
30	4.6061	0	0	0
31	4.9263	0	0	0
32	5.2582	0	0	0
33	5.6019	0	0	0
34	5.9575	0	0	0
35	6.3250	0	0	0
36	6.7048	0	0	0
37	7.0968	0	0	0
38	7.5014	0	0	0
39	7.9185	0	0	0
40	8.3485	0	0	0
41	8.7914	0	0	0
42	9.2476	0	0	0
43	9.7171	0	0	0
44	10.2002	0	0	0
45	10.6971	0	0	0
46	11.2081	0	0	0
47	11.7334	0	0	0
48	12.2732	0	0	0
49	12.8278	0	0	0
50	13.3975	0	0	0
51	13.9826	0	0	0
52	14.5834	0	0	0
53	15.2002	0	0	0
54	15.8335	0	0	0
55	16.4835	0	0	0
56	17.1507	0	0	0
57	17.8355	0	0	0
58	18.5384	0	0	0
59	19.2597	0	0	0
60	20.0000	0	0	0
61	20.7599	0	0	0
62	21.5398	0	0	0
63	22.3405	0	0	0
64	23.1625	0	0	0
65	24.0066	0	0	0

1. The second derivative of z with respect to s

Table D.3: The Road Data File that was Used in Example 5 (Continued)

x	y	z	zp_s^1	α
66	24.8734	0	0	0
67	25.7639	0	0	0
68	26.6788	0	0	0
69	27.6191	0	0	0
70	28.5857	0	0	0
71	29.5798	0	0	0
72	30.6026	0	0	0
73	31.6553	0	0	0
74	32.7393	0	0	0
75	33.8562	0	0	0
76	35.0077	0	0	0
77	36.1956	0	0	0
78	37.4220	0	0	0
79	38.6893	0	0	0
80	40.0000	0	0	0
81	41.3570	0	0	0
82	42.7636	0	0	0
83	44.2237	0	0	0
84	45.7414	0	0	0
85	47.3217	0	0	0
86	48.9706	0	0	0
87	50.6948	0	0	0
88	52.5026	0	0	0
89	54.4039	0	0	0
90	56.4110	0	0	0
91	58.5392	0	0	0
92	60.8082	0	0	0
93	63.2440	0	0	0
94	65.8826	0	0	0
95	68.7750	0	0	0
96	72.0000	0	0	0
97	75.6895	0	0	0
98	80.1003	0	0	0
99	85.8933	0	0	0
100	100.0000	0	0	0

1. The second derivative of z with respect to s

Table D.4: The Road Data File that was Used in Example 5 (Continued)

x	y	z	zp_s^1	α
0	0.0000	0	0	0
1	0.0000	0	0	0
100	0.0000	0	0	0
160	0.0000	5.25	0	0
200	0.0000	5.25	0	0
200.5	0.0025	5.25	0	0
201	0.0100	5.25	0	0
201.5	0.0225	5.25	0	0
202	0.0400	5.25	0	0
202.5	0.0625	5.25	0	0
203	0.0901	5.25	0	0
203.5	0.1227	5.25	0	0
204	0.1603	5.25	0	0
204.5	0.2029	5.25	0	0
205	0.2506	5.25	0	0
205.5	0.3034	5.25	0	0
206	0.3613	5.25	0	0
206.5	0.4243	5.25	0	0
207	0.4924	5.25	0	0
207.5	0.5657	5.25	0	0
208	0.6441	5.25	0	0
208.5	0.7278	5.25	0	0
209	0.8167	5.25	0	0
209.5	0.9108	5.25	0	0
210	1.0102	5.25	0	0
210.5	1.1149	5.25	0	0
211	1.2250	5.25	0	0
211.5	1.3405	5.25	0	0
212	1.4614	5.25	0	0
212.5	1.5877	5.25	0	0
213	1.7196	5.25	0	0
213.5	1.8570	5.25	0	0
214	2.0000	5.25	0	0
214.5	2.1487	5.25	0	0
215	2.3030	5.25	0	0
215.5	2.4632	5.25	0	0
216	2.6291	5.25	0	0
216.5	2.8010	5.25	0	0
217	2.9787	5.25	0	0
217.5	3.1625	5.25	0	0
218	3.3524	5.25	0	0

1. The second derivative of z with respect to s

Table D.5: The Road Data File with the Centerline of the Road Specified that was Used in Example 6

x	y	z	zp_s^1	α
218.5	3.5484	5.25	0	0
219	3.7507	5.25	0	0
219.5	3.9593	5.25	0	0
220	4.1742	5.25	0	0
220.5	4.3957	5.25	0	0
221	4.6238	5.25	0	0
221.5	4.8586	5.25	0	0
222	5.1001	5.25	0	0
222.5	5.3486	5.25	0	0
223	5.6041	5.25	0	0
223.5	5.8667	5.25	0	0
224	6.1366	5.25	0	0
224.5	6.4139	5.25	0	0
225	6.6987	5.25	0	0
225.5	6.9913	5.25	0	0
226	7.2917	5.25	0	0
226.5	7.6001	5.25	0	0
227	7.9167	5.25	0	0
227.5	8.2418	5.25	0	0
228	8.5754	5.25	0	0
228.5	8.9178	5.25	0	0
229	9.2692	5.25	0	0
229.5	9.6298	5.25	0	0
230	10.0000	5.25	0	0
230.5	10.3799	5.25	0	0
231	10.7699	5.25	0	0
231.5	11.1702	5.25	0	0
232	11.5813	5.25	0	0
232.5	12.0033	5.25	0	0
233	12.4367	5.25	0	0
233.5	12.8819	5.25	0	0
234	13.3394	5.25	0	0
234.5	13.8095	5.25	0	0
235	14.2929	5.25	0	0
235.5	14.7899	5.25	0	0
236	15.3013	5.25	0	0
236.5	15.8276	5.25	0	0
237	16.3697	5.25	0	0
237.5	16.9281	5.25	0	0
238	17.5038	5.25	0	0

1. The second derivative of z with respect to s

Table D.6: The Road Data File with the Centerline of the Road Specified that was Used in Example 6 (Continued)

x	y	z	zp_s^1	α
238.5	18.0978	5.25	0	0
239	18.7110	5.25	0	0
239.5	19.3447	5.25	0	0
240	20.0000	5.25	0	0
240.5	20.6785	5.25	0	0
241	21.3818	5.25	0	0
241.5	22.1118	5.25	0	0
242	22.8707	5.25	0	0
242.5	23.6609	5.25	0	0
243	24.4853	5.25	0	0
243.5	25.3474	5.25	0	0
244	26.2513	5.25	0	0
244.5	27.2020	5.25	0	0
245	28.2055	5.25	0	0
245.5	29.2696	5.25	0	0
246	30.4041	5.25	0	0
246.5	31.6220	5.25	0	0
247	32.9413	5.25	0	0
247.5	34.3875	5.25	0	0
248	36.0000	5.25	0	0
248.5	37.8448	5.25	0	0
249	40.0501	5.25	0	0
249.5	42.9466	5.25	0	0
250	50.0000	5.25	0	0

1. The second derivative of z with respect to s

Table D.7: The Road Data File with the Centerline of the Road Specified that was Used in Example 6 (Continued)

x	z	zp_x^1	α
0	0	0	0
100	0	0	0
160	5.25	0	0
250	5.25	0	0

1. The second derivative of z with respect to x

Table D.8: The Road Data File with the Road Plane Specified that was Used in Example 6